Approximation Algorithms for Network Design Problems with Node Weights

C. Chekuri* M. T. Hajiaghayi† G. Kortsarz‡ M. R. Salavatipour§

Abstract

We consider several fundamental network design problems in undirected graphs with weights (costs) on both *edges and nodes*. We obtain the first poly-logarithmic approximation ratios for some classical problems that have been studied in the edge-weighted setting. Our main results are the following.

- An $O(\log^2 n)$ approximation for the Steiner forest problem.
- An $O(\log n)$ approximation for the single-sink non-uniform buy-at-bulk network design.
- An $O(\min(\log^3 n \log D, \log^5 n \log\log n))$ approximation for the node-weighted multicommodity non-uniform buy-at-bulk network design, where $D = \sum_i \delta_i$ is the total demand.

We believe that the problems and results are of interest from both theoretical and practical points of view.

*Department of Computer Science, University of Illinois, Urbana, IL 61801. Email: chekuri@cs.uiuc.edu. Part of this work was done while the author was at Lucent Bell Labs.

†Department of Computer Science, Carnegie Mellon University. Email: hajiagha@cs.cmu.edu.

‡Department of Computer Science, Rutgers University-Camden. Email: guyk@crab.rutgers.edu.

§Department of Computing Science, University of Alberta. Email: mreza@cs.ualberta.ca. Supported by NSERC grant No. G121210990, and a faculty start-up grant from University of Alberta.

1 Introduction

Network design problems involve finding a minimum cost (sub) network that satisfies various properties, often involving connectivity. Simple examples include spanning trees, Steiner trees, and k-connected subgraphs. These problems are of fundamental importance in combinatorial optimization and also arise in a number of applications in computer science and operations research. The cost in a typical network design problem is some function of the chosen *edges*. In this paper we consider network design problems with costs (or weights) on both *edges* and *nodes* of the graph. We are motivated by both theoretical as well as practical considerations. The node-weighted problems are a natural generalization of the edge weighted problems (weights on edges can be translated to weights on nodes in an easy fashion). It is often possible to reduce the node-weighted problem to a corresponding edge-weighed problem, however this requires making the graph directed. Problems on directed graphs are typically more complex (harder to approximate for instance) than the ones in undirected graphs and hence it is desirable to work directly on node-weighted problems in undirected graphs. Node weights also arise naturally in a number of practical applications. For example, in telecommunications, expensive equipment such as routers and switches are at the nodes of the underlying network and it is natural to model some of these problems as node-weighted problems. Often, these node weights are translated into edge weights in an approximate fashion since the edge-weighted problems are better understood. Apart from the work of Klein and Ravi [26] who gave an $O(\log n)$ approximation algorithm for the node-weighted Steiner tree problem, and some subsequent results on variants of this problem obtained by Guha et al. [18] and Moss and Rabani [28], there does not appear to be much literature on node weighted variants[1]. In this paper we consider the node-weighted versions of some well studied network design problems and obtain the first non-trivial approximation algorithms. We hope that our results will spur further work. We now describe the problems we study and the results we obtain. The corresponding edge-weighted problems are well studied.

Node-weighted Steiner forest: We are given an undirected graph $G = (V, E)$ and a non-negative cost function $c : V \rightarrow \mathcal{R}^+$ on the nodes of the graph. We are also given h node pairs $s_1t_1, s_2t_2, \ldots, s_ht_h$ and the goal is to find a minimum cost subset of nodes A such that each pair s_it_i is connected in the subgraph $G[A]$ induced by A.

Node-weighted Non-uniform Buy-at-Bulk: This problem generalizes the Steiner forest problem. We are given an undirected graph G and node pairs $s_1t_1, s_2t_2, \ldots, s_ht_h$ as before. The pairs have non-negative demands; δ_i is the demand for pair s_it_i. Each node $v \in V$ has a monotone sub-additive real valued function $f_v : \mathcal{R}^+ \rightarrow \mathcal{R}^+$ associated with it. A feasible solution consists of a multi-commodity flow for the pairs in which δ_i flow is routed between s_i and t_i. The cost of the flow is $\sum_{v \in V} f_v(x_v)$ where x_v is the total flow that is routed through a node v (We assume that a flow that originates at a node v is also routed through v). The objective is to find a feasible routing (or flow) for the pairs that minimizes the cost. We obtain a single-sink (or equivalently single-source) problem if all pairs share a common sink node, that is, the pairs are of the form $st_1, st_2, \ldots, st_k$ with s as the sink. We obtain a *uniform* instance if there is a function f such that for each v, $f_v = c_v f$ for some cost function $c : V \rightarrow \mathcal{R}^+$.

Results: We obtain the first poly-logarithmic approximation ratios for the two problems we consider. These are also the first non-trivial results for these problems.

- An $O(\log^2 h)$ approximation for the node-weighted Steiner forest problem (Theorem 3.3).
- An $O(\log h)$ approximation and integrality gap for the node-weighted single-sink buy-at-bulk network design problem (Theorem 4.1). Unless $P = NP$ the ratio we obtain is tight up to a constant factor.
- An $O(\min(\log^3 h \log D, \log^5 h \log\log h))$ approximation for the multi-commodity buy-at-bulk problem where $D = \sum_i \delta_i$ is the total demand (Theorem 5.1).

Remark:

Our approximation for the Steiner forest is in fact $O(\beta \cdot \log n)$ with β the best known approximation for the budgeted coverage with node weights (BCNW) (see [18, 28]). A graph $G(V, E)$ with vertex costs and a set

[1]There is literature on node capacitated routing problems (see e.g. [9, 13, 21] for some recent works) but those usually do not involve cost minimization.

$T \subseteq V$ of terminals, and a budget B are given. The goal is to choose a subtree Q of G with budget at most B maximizing the number $|Q \cap T|$ of terminals in Q. In [28] an $O(\log n)$ approximation is designed for this problem. It is conjectured that the problem admits an $O(1)$ approximation. If the conjecture hold true, then our algorithm can be used to derive an $O(\log n)$ approximation for the vertex-costs Steiner forest problem. Even the case of all equal sinks, namely, the vertex costs Steiner *tree* problem is hard to approximate within $c \ln n$ for some $c > 1$ (see [26]). Therefore, it may be possible to use our algorithm to derive an approximation algorithm that (if $P \neq NP$) is optimal up to constants approximation

Related Work: Network design problems are of fundamental importance in combinatorial optimization and there is a vast literature on problems and results. We refer the reader to [30, 15] for classical results on polynomial time algorithms and to [16, 23, 32, 24, 14] for results on approximation algorithms. Here we briefly discuss the known results and techniques for some specific problems that are closely related to the problems we consider.

For the edge-weighted Steiner forest problem a $(2 - 1/h)$-approximation is known, first shown via a primal-dual algorithm by Aggarwal, Klein and Ravi [1]. The primal-dual approach has been generalized to other connectivity functions by Goemans and Williamson [16], and Jain [24] gave a 2-approximation for a large class using iterative rounding. The node-weighted version behaves differently. Klein and Ravi [26] showed, via an easy reduction from the set cover problem, that the node weighted Steiner tree problem is $\Omega(\log h)$-hard to approximate unless $P = NP$. They also obtained a matching approximation ratio using a greedy merging algorithm. In a subsequent work Guha et al. [18] showed that a natural LP relaxation has an $O(\log h)$ integrality gap and used this to prove an $O(\log^2 h)$ approximation for a quota version of the problem. Moss and Rabani [28] improved the ratio for the quota version to $O(\log h)$ and also proved a stronger Lagrangian multiplier property for the LP relaxation. Surprisingly no non-trivial result is known for the node-weighted Steiner forest problem.

The buy-at-bulk network design problem is motivated by economies of scale that arise in a number of applications, most notably in telecommunications. This problem is studied as the fixed charge network flow problem in operations research. Approximation algorithms, starting with the work of Salman et al. [29], have been of much interest. All the known results are only for the edge-weighted problems. For the uniform case Awerbuch and Azar [5] gave a reduction to the problem of approximating a finite metric via random tree metrics and this results in an $O(\log n)$-approximation using [12]. An improved $O(1)$-approximation is obtained for the *uniform* single-sink case first by Guha, Meyerson and Munagala [17] with further improvements and simplifications [31, 20]. A special case of the multicommodity flow version, known as the rent-or-buy problem also admits a constant factor approximation [19]. The non-uniform versions of the problem turn out to be harder. For the single-sink case, Meyerson, Munagala and Plotkin [27] obtained an $O(\log h)$-approximation. Their randomized algorithm was derandomized via an LP relaxation in [8]. For the multi-commodity problem the first non-trivial result is due to Charikar and Karagiazova [6] who obtained an $O(\log D \exp(O(\sqrt{\log h \log\log h})))$-approximation. Very recently, the authors of this paper obtained a first poly-logarithmic approximation [7]. The ratio obtained is $O(\min(\log^3 h \log D, \log^5 h \log\log h))$. Andrews [3] showed an $\Omega(\log^{1/4-\epsilon} n)$-hardness for the uniform case and an $\Omega(\log^{1/2-\epsilon} n)$-hardness for the non-uniform case, both under the assumption that $NP \not\subseteq ZTIME(n^{\text{polylog}(n)})$. For the non-uniform single-sink case Chuzhoy et al. [10] showed an $\Omega(\log\log n)$-hardness of approximation under the assumption that $NP \not\subseteq DTIME(n^{\log\log\log n})$. We note that the inapproximability results for the edge-weighted problems apply to their node-weighted generalizations.

Techniques: The results in this paper build upon several ideas from the related work we described above. For the Steiner forest problem we use the greedy approach inspired by our earlier work [22] and use the algorithm of Moss and Rabani [28] for the quota version of the Steiner tree problem as a black box. For the buy-at-bulk problems, we use ideas from our recent work on the edge-weighted non-uniform buy-at-bulk problem [7] to reduce a multi-commodity problem to essentially a variant of its corresponding single-sink problem. In [7] two approaches, one based on a greedy algorithm, and the other based on an LP relaxation are used to solve the single-sink problem. We obtain an algorithm for the single-sink case by building upon and generalizing ideas in several papers [26, 27, 8].

Organization: Several technical details including the reduction of the buy-at-bulk problem to the two-cost network design problem are described in Section 2. We start with the $O(\log^2 n)$-approximation algorithm for the node-weighted Steiner forest in Section 3. In Section 4, we present the $O(\log n)$-approximation for node-weighted non-uniform single-sink buy-at-bulk problem which is interestingly tight (we currently lack such a tight result for

the edge-weighted case). Finally, we present approximation algorithm for node-weighted non-uniform multicommodity buy-at-bulk in Section 5.

2 Preliminaries

All graphs we consider are undirected. Nodes and vertices are used interchangeably. As mentioned earlier, we consider the settings in which both *edges* and *vertices* have weights. However, we can easily transform this settings into one in which only the vertices have weights by subdividing every edge (i.e. replacing it with a path of length 2) and giving the new vertex the weight equal to the the weight of original edge. Using the same transformation, it is easy to see we can reduce the edge-weighted version of all the problems mentioned earlier to the node-weighted.

Recall that in the node-weighted non-Uniform buy-at-bulk problem, we are given an undirected graph $G = (V, E)$ on n vertices, a monotone sub-additive cost function $f_v : \mathcal{R}^+ \rightarrow \mathcal{R}^+$, and a set of h demand pairs $\mathcal{T} = \{s_1t_1, s_2t_2, \ldots, s_ht_h\}$, where δ_i is the demand for pair s_it_i. consists of which solution consists of a subgraph $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq G[V']$ with non-zero flow and a routing of flow δ_i for each pair s_it_i in G'. As shown in earlier works [4, 27, 7], we can approximate each function f_v by a collection of simple piece-wise linear functions of the form $a + bx$. Then for each vertex v with cost function f_v, we create several copies of v all having the same set of neighbors as v and each having one of the linear cost functions $a + bx$. This allows us to reformulate the buy-at-bulk network design problem as a *two-cost* network design problem, at a loss of factor 2 in the approximation ratio. In this setting, an instance of node-weighted non-uniform multicommodity buy-at-bulk (NMC-BB) consists of a graph G and demand pairs $\mathcal{T} = \{s_1t_1, s_2t_2, \ldots, s_ht_h\}$. Each $s_i, t_i \in V$ has a demand $\delta_i \geq 0$. We are given two separate functions $c : V \rightarrow \mathcal{R}^+$ and $\ell : V \rightarrow \mathcal{R}^+$; we call c_v and ℓ_v the *cost* and *length* of v, respectively. We think of c_v as the fixed cost of v and ℓ_v as the incremental or flow-cost of v. The goal is to find a minimum cost feasible solution where a feasible solution consists of a subset of nodes $V' \subseteq V$ that includes all the terminals. The subset V' implicitly specifies the induced subgraph $G' = G[V']$. The cost of the solution specified by V' is given as

$$c(V') + \sum_{i=1}^{h} \delta_i \cdot \ell_{G'}(s_i, t_i), \tag{1}$$

where $c(V') = \sum_{v \in V'} c_v$ and $\ell_{G'}(u, v)$ is the shortest ℓ-node weighted path distance between u and v in G'. If all the sinks are the same we have the node-weighted non-uniform single-sink buy-at-bulk (NSS-BB) problem. We are given an an undirected graph $G = (V, E)$ with a designated root vertex r, a set of terminals $T \subseteq V$, a demand function $\delta : T \cup \{r\} \rightarrow \mathbb{R}^+$, a cost function $c : V \rightarrow \mathbb{R}^+$, and a length function $\ell : V \rightarrow \mathbb{R}^+$. A feasible solution is a connected subgraph F containing r and spanning all the terminals.

Our goal is to route $\delta(t)$ units of flow from each terminal t to root r. In the rest of the paper, we restrict our attention to the two-cost network design formulation of NMC-BB and NSS-BB.

Our algorithms for the node-weighted Steiner forest and node-weighted non-uniform multicommodity buy-at-bulk have a greedy structure; iteratively we try to find a partial solution (a solution that connects/routes some of the remaining pairs) at low density, where the density is the cost of the partial solution over the number of new pairs it connects/routes. We will use the following basic lemma in the analysis of these algorithms (see e.g., [25]).

Lemma 2.1 *Suppose that an algorithm works in iterations and in iteration i it finds a partial solution $V_i \subseteq V$ that routes a new subset T_i of the demands. Let* OPT *be the cost of the optimum solution and u_i be the number of unrouted demands at the time V_i is found. If for every i, the cost of the partial solution $G[V_i]$ over the number of pairs it routes is at most $f(n) \cdot \frac{\text{OPT}}{u_i}$, then the cost of the solution returned by the algorithm is at most $f(n) \cdot (\ln n + 1) \cdot$ OPT.*

3 Approximation Algorithm for Node-Weighted Steiner Forest

Recall that the instance of node-weighted Steiner forest problem consist of an undirected graph $G = (V, E)$, a cost function $c : V \rightarrow \mathbb{R}^+$, and a collection of pairs of source-sinks $\mathcal{T} = \{(s_1, t_1), \ldots, (s_h, t_h)\}$ with $s_i, t_i \in V$. Let us first start with a definition which is basic to our algorithm in this section.

In the *(Rooted) Density node-weighted Steiner tree* problem, given an undirected graph $G = (V, E)$, a cost function $c : V \to \mathbb{R}^+$, a root $r \in V$, and a set of terminals $T \subseteq V$ containing the root r, our goal is to find a tree F rooted at r such that $\sum_{v \in F} c(v)/|F \cap T|$ is minimized. Our main result of this section is the following.

Theorem 3.1 *An α-approximation algorithm for the (rooted) density node-weighted Steiner tree problem implies an $O(\alpha \log n)$-approximation algorithm for node-weighted Steiner forest problem.*

The (rooted) density node-weighted Steiner tree is closely related to the following Budget-constrained coverage with node weights (BCNW) problem [28]. Given a node-weighted graph G, budget B and set $T \subseteq V$ of terminals, find a tree of cost at most B that contains the maximum possible number of terminals. In [28] an $O(\log n)$ ratio approximation is given for BCNW and conjectured that BCNW admits an $O(1)$ ratio approximation.

Lemma 3.2 *If there is an α-approximation for BCNW then there is a 2α-approximation for rooted density node-weighted Steiner tree.*

Proof. Let α be the best possible ratio approximation for BCNW. Given an instance of (rooted) density node-weighted Steiner tree, we can guess (e.g. try all possible values for) the cost B for the optimal solution. Then we remove all the vertices of distance more than B from root r. The best tree of budget B is approximately found using the α-approximation for BCNW, and then r is joined with a path of cost at most B to an arbitrary vertex in the computed tree. This affects the ratio by a factor of at most 2. □

From Theorem 3.1, Lemma 3.2, and [28] which proves that there is an $O(\log n)$-approximation for BCNW, it follows that:

Theorem 3.3 *There is an $O(\log^2 n)$-approximation algorithm for the node-weighted Steiner forest problem.*

It is NP-hard to approximate the node-weighted Steiner forest problem within a factor of $c \log n$ for some absolute constant $c > 0$. Therefore, if the conjecture of [28] is true, our algorithm can be used to derive an approximation ratio for the Steiner forest problem that is optimal up to a constant factor. In the rest of this section we prove Theorem 3.1. The following assumption is useful.

Assumption 3.4 *Each source and sink has degree 1 in G.*

The assumption is easily satisfied by hanging a degree one vertex of cost 0 from each terminal and making this set of new nodes to be the terminals.

The algorithm for Theorem 3.1 is a greedy algorithm which uses ideas from the greedy algorithm for multicommodity buy-at-bulk [7]. It runs in iterations and uses the α-approximation algorithm for (rooted) density node-weighted Steiner tree as a subroutine. We refer to this subroutine by D-NWST. At each iteration, the algorithm finds a partial solution that connects some of the terminals (source-sink pairs) that are not already connected using D-NWST. Then these newly connected terminals are removed. A partial solution is defined by a subset of nodes and the subgraph induced by them. The density of the partial solution is the ratio of the cost of the partial solution to the number of new pairs it connects. Let OPT be an optimum solution for the remaining pairs (that are not connected yet) and OPT be its cost. Clearly OPT is at most the cost of a global optimum solution since at any given time we can add the vertices of the global optimum solution to cover all the remaining pairs. Assuming that the density of every partial solution is at most $O(\alpha \cdot \text{OPT}/h')$, where h' is the number of remaining pairs, using an standard set-cover type analysis, this scheme yields an $O(\alpha \log n)$ approximation for the node-weighted Steiner forest problem.

The main procedure in our algorithm is a procedure called Partial which finds a good density partial solution. Each iteration of the algorithm makes a call to Partial until all the pairs are connected. Consider one such iteration and let $\mathcal{T}'$ be the set of remaining (not connected) pairs of the original instance and $h' = |\mathcal{T}'|$. The procedure Partial works in rounds and each round contains two phases: sources phase and sinks phase. In the sources phase (which runs in iterations) we start from one of the remaining sources, say s, as the root and consider s together with other remaining sources as terminals. In each iteration we try to find a low density Steiner tree rooted at

s. We do this by calling the subroutine D-NWST. If the density of the tree returned by D-NWST is at most $8\alpha.\text{OPT}/h'$ then we contract all the nodes of that tree into s, set the cost of s to zero (because we have already paid for it), and continue with the next iteration. Since sinks have degree 1, sinks are never contracted into s in the sources phase. Indeed, sinks cannot help s reach any source and may be ignored in the source phase. If no low density subtree can be found, the sources phase terminates. At this stage, let X be the set of sources that are contracted into s.

We define the set of terminals Y to be the sinks that correspond to a source in X. Now we start the sinks phase that like the sources phase runs in iterations. In each iteration we call subroutine D-NWST with s being the root and the sinks in Y being the terminals to find the best density Steiner tree. Each time we find a tree, contract all of its nodes into s and set the cost of s to zero. We repeat this until we have at least $|X|/32$ sinks corresponding to the sources of X contracted into s. At the end if the density of the connected graph induced by the set of the nodes contracted into s is small we return the set of nodes contracted into s as the partial solution. Otherwise, if the resulting tree has a too large density we (temporarily) discard all the sources in X and their corresponding sinks and start over the next round of Partial restricted to non-discarded pairs. A key claim we need to prove is that before the set of non-discarded pairs turns empty, some new pairs are covered. For a set of nodes F, let $T(F)$ be the set of terminals in F and $c(F)$ denote the total cost of the nodes in F. Let $\mathcal{T}'$ be the set of remaining (not connected) pairs of the original instance. See Figure 2 in Appendix for the detailed description of this procedure.

Recall that OPT is a collection of disjoint trees $\mathcal{S} = \{G_1, \ldots, G_\ell\}$. Let $\mathcal{T}(G_i)$ be the set of pairs routed in G_i. By deleting every tree G_i whose density is larger than $2\text{OPT}/h'$, we obtain:

Lemma 3.5 *There is a collection $\mathcal{S}' = \{G_1, \ldots, G_{\ell'}\}$ of disjoint trees of $\mathcal{S}$ such that:*

1. *Together they contain at least $\lceil h'/2 \rceil$ of the pairs*
2. *Density of each $G_i \in \mathcal{S}'$ is at most $2\text{OPT}/h'$.*

We need a few definitions for the analysis of the algorithm. A tree in $\mathcal{S}'$ from which at most a fraction of $1/4$ of the pairs are discarded by Partial is called a *good tree*. Note that a good tree may become bad during the course of the algorithm as some of its source-sink pairs get discarded. A source-sink pair is called a *good pair* if it belongs to a good tree at the time the source is being considered by the algorithm. A source (sink) is a good source (sink) if it belongs to a good pair. Others are called bad pairs (or bad sources). Consider V_s (the set of nodes contracted into s) at the end of the sources-phase. One round of Partial is called a *good round* if at least a fraction of $1/16$ of the sources of V_s are from good pairs, i.e. X contains at least $\lceil x/16 \rceil$ good sources. Other rounds are called bad rounds.

Lemma 3.6 *In every call to Partial, there is at least one good round before all the pairs are discarded.*

Proof. By way of contradiction, suppose that all the rounds are bad and the algorithm continues until all the pairs are discarded by the procedure. Let k_i be the number of pairs discarded in round i. This implies that $\sum_i k_i = h'$. By Lemma 3.5, the number of sources (pairs) in $\mathcal{S}'$ is at least $\lceil h'/2 \rceil$. Initially, all the sources in $\mathcal{S}'$ are good. By definition, for every tree $G_i \in \mathcal{S}'$, the first $\lceil |\mathcal{T}(G_i)|/4 \rceil$ sources of G_i that are discarded are good and then the rest become bad sources. Therefore, the number of good sources that become bad is at most 3 times the number of good sources that are discarded. Therefore, the total number of good sources discarded and the number of good sources that become bad is at most $\sum_i 4\lfloor k_i/16 \rfloor \leq \lfloor h'/4 \rfloor$. So using Lemma 3.5, there are at least $\lceil h'/4 \rceil \geq 1$ good pairs left and so we cannot have discarded all the pairs. Hence there must be a good round. □

From now on, we focus on a good round and let $\mathcal{S}'' \subseteq \mathcal{S}'$ be the set of good trees of $\mathcal{S}'$ that have at least one vertex in V_s. In the next lemma, the term "number of original sources" refers to the situation at the beginning (before any source is discarded).

Lemma 3.7 *For every tree $G_i \in \mathcal{S}''$, if V_s contains a vertex of G_i then V_s contains at least half of the original sources of G_i.*

Proof. By way of contradiction, assume that there is a tree $G_i \in \mathcal{S}''$ which has a vertex in V_s but less than $1/2$ of the sources of G_i are in V_s. This means that by the end of last successful iteration of the sources-phase, root s belongs to G_i and more than $\lceil |\mathcal{T}(G_i)|/4 \rceil$ of the sources of G_i are still left (uncontracted into s). This is because G_i is a good tree and therefore at most $\lfloor |\mathcal{T}(G_i)|/4 \rfloor$ of them are discarded; thus at least $\lceil 3|\mathcal{T}(G_i)|/4 \rceil$ sources are not discarded. As by assumption less than $1/2$ of the sources of $\mathcal{T}(G_i)$ belong to s, more than $\lceil |\mathcal{T}(G_i)/4 \rceil$ do not belong to s nor are discarded, and so are available for choice. Thus, there is a tree F_s rooted at s and containing at least $\lceil |\mathcal{T}(G_i)|/4 \rceil$ (new) sources at cost at most $c(G_i)$ (this can be obtained by selecting all the vertices and edges of G_i not contracted into s). The density of F_s is at most $4c(G_i)/|\mathcal{T}(G_i)| \leq 8\text{OPT}/h'$. Since D-NWST is an α-approximation, in the last (unsuccessful) iteration of the sources-phase we would have found a tree F_s with density at most $8\alpha.\text{OPT}/h'$ and so the sources-phase should have not terminated, a contradiction. □

Lemma 3.7 implies that V_s (at the end of sources-phase in a good round) contains at least half the total number of sources of all the trees in $\mathcal{S}''$. Consider a good round and let $x = |X|$ be the number of sources contracted into s in the sources-phase. If $c(V_s)$ is the cost of nodes in V_s after the sources-phase then

$$c(V_s) \leq 8\alpha \cdot x \cdot \text{OPT}/h'. \tag{2}$$

If in the sinks-phase we define V_t to consist of the rest of the vertices of the trees in $\mathcal{S}''$, we can connect at least $\lceil x/16 \rceil$ source-sink pairs (those good pairs whose sources are in X) using $V_s \cup V_t$. Since the density of each tree in $\mathcal{S}''$ is at most $2\text{OPT}/h'$ and $x \geq |\mathcal{T}(\mathcal{S}'')|/2$ (by Lemma 3.7), the total cost of the nodes in $\mathcal{S}''$ is

$$c(\mathcal{S}'') \leq 4x \cdot \text{OPT}/h'. \tag{3}$$

Thus at the beginning of the sinks-phase, there exists a tree F_t rooted at s over the terminals in Y with cost at most $c(\mathcal{S}'')$ and containing at least $\lceil x/16 \rceil$ terminals, i.e. has density at most $64\text{OPT}/h'$. We show below that the sinks-phase finds a collection of Steiner trees rooted at s over terminals Y with a total number of terminals at least $\lceil x/32 \rceil$ each of which has density at most $128\alpha \cdot \text{OPT}/h'$.

Lemma 3.8 *In a good round, at the end of the sinks-phase set $V_s \cup V_t$ induces a subgraph (partial solution) that has density at most $384\alpha \cdot \text{OPT}/h'$.*

Proof. Suppose that we have $q \geq 1$ iterations of the repeat loop in the sinks-phase and we find Steiner trees $F_t^1, \ldots, F_t^q$ whose number of terminals are $x_1, \ldots, x_q$, respectively, where $\sum_{1 \leq i \leq q} x_i \geq \lceil x/32 \rceil$ but $\sum_{1 \leq i < q} x_i < \lceil x/32 \rceil$. We prove by induction that the density of each of $F_t^1, \ldots, F_t^q$ is at most $128\alpha.\text{OPT}/h'$. By the discussion above, at the beginning of the sinks-phase, there exists a tree F_t with density at most $c(\mathcal{S}'')/\lceil x/16 \rceil \leq 64\text{OPT}/h'$. Because D-NWST is an α-approximation, the density of F_t^1 is at most $64\alpha \cdot \text{OPT}/h'$. For the induction step, suppose we have found trees $F_t^1, \ldots, F_t^i$, for some $1 \leq i < q$. So we have contracted all the vertices of these trees into s. As $\sum_{1 \leq i < q} x_i < \lceil x/32 \rceil$ and F_t has at least $\lceil x/16 \rceil$ (good) terminals of Y, there are at least $\lceil x/32 \rceil$ terminals of tree F_t left, and clearly these can be connected to s at cost at most $c(\mathcal{S}'')$. Thus, there is a Steiner tree rooted at s with density at most $c(\mathcal{S}'')/\lceil x/32 \rceil \leq 128\text{OPT}/h'$. Hence, F_t^{i+1} has density at most $128\alpha \cdot \text{OPT}/h'$, as wanted.

Since we repeat the sinks-phase until there are at least $\lceil x/32 \rceil$ sinks corresponding to the sources of X contracted into s, at the end of sinks-phase, we have at least $\lceil x/32 \rceil$ pairs contracted into s. Thus $V_s \cup V_t$ connect at least $\lceil x/32 \rceil \geq 1$ pairs. The total cost of the partial solution at the end of sinks-phase (i.e. cost of $V_s \cup V_t$) is at most $c(V_s)$ plus the cost of the trees found in the sinks-phase. By Equation (2): $c(V_s) \leq 8\alpha \cdot x \cdot \text{OPT}/h'$. The cost of the trees found in the sinks-phase is at most $\alpha \cdot c(\mathcal{S}'')$ which by Equation (3) is at most $4\alpha \cdot x \cdot \text{OPT}/h'$. Thus the total cost is at most $12\alpha \cdot x \cdot \text{OPT}/h'$ and because it has at least $\lceil x/32 \rceil$ pairs, the density is at most $(12 \times 32)\alpha \cdot \text{OPT}/h'$. □

Proof of Theorem 3.1. Follows from Lemma 3.8 and the fact that there is at least one good round in every call to Partial (by Lemma 3.6), together with Lemma 2.1. □

4 Node-Weighted Non-Uniform Single-Sink Buy-at-Bulk

Recall that the instance to NSS-BB is an undirected graph $G = (V, E)$ with a designated root vertex r, a set of terminals $T \subseteq V$, a demand function $\delta : T \cup \{r\} \to \mathbb{R}^+$, a cost function $c : V \to \mathbb{R}^+$, and a length function $\ell : V \to \mathbb{R}^+$. Our main result of this section is:

Theorem 4.1 *There is a deterministic $O(\log h)$-approximation algorithm for NSS-BB where h is the number of terminals. In fact, we find a solution that is within a factor $O(\log h)$ of the optimal solution to a standard LP relaxation.*

It is easy to see that if we have costs and lengths on both V and E we can reduce this to the node-weighted case. If for a vertex v, $c(v) = \ell(v) = 0$ then we can add this vertex to any solution at no costs. So, without loss of generality, we may further assume that for every vertex v, either $c(v) > 0$ or $\ell(v) > 0$. For technical reasons we assume that $\delta(r)$ is the largest demand and $c(r) = \ell(r) = 0$.

Since NSS-BB generalizes the node-weighted Steiner tree, which has a $\Omega(\log n)$-hardness [26] (via a simple reduction from set-cover):

Corollary 4.2 *NSS-BB has an approximability threshold of $\Theta(\log n)$, unless $\mathrm{P} = \mathrm{NP}$.*

The algorithm for Theorem 4.1 uses ideas from the works of Klein and Ravi [26], Guha et al. [18] and Chekuri et al. [8] and Meyerson et al. [27]. In particular, we use the spider ideas from [26] and randomized merging from [27].

A *spider* is a connected graph with at most one vertex of degree more than two. So we can think of it as a tree that consists of some paths all of which are sharing exactly one of their end-points. The *center* of a spider is a node from which there are edge-disjoint paths to the leaves of the spider. So if the spider has a vertex of degree at least three, its center is unique. Every leaf of a spider must be a terminal. The density of a spider is the ratio of its total cost over the number of terminals in the union of its leaves and its center, where the total cost depends on the problem definition. For the problem of node-weighted Steiner tree (i.e. when we do not have a length function) the total cost is just the sum of the weights of the nodes in the spider. For this problem, Klein and Ravi [26] showed the existence of a decomposition of the optimum solution into spiders. Therefore, there is always an spider whose density is no more than the density of the optimum (which is the cost of the optimum over the total number of terminals). They also show how to find a best density spider in polynomial time. Given this tool in hand, we can iteratively find the best density spider and contract all the nodes in that into a single node, until all the terminals are contracted into r. Again, using a standard set-cover type analysis, this yields an $O(\log n)$-approximation for the node-weighted Steiner tree. Given that the set-cover can be reduced to node-weighted Steiner tree problem, we have an $\Omega(\log n)$-hardness too; so $\Theta(\log n)$ is the approximability threshold for node-weighted Steiner tree. Guha et al. [18] later showed that in fact the density of the best density spider is no more than the density of the optimum *fractional* solution.

Overview of the algorithm for NSS-BB: Our algorithm for Theorem 4.1 has a similar structure. Since we will compare the ratio of our algorithm against the optimum fractional solution let us first formulate NSS-BB as an IP for which we have the following LP relaxation. For $t \in T$, let $\mathcal{P}_t$ denotes the set of directed paths from root r to t. We assume that the terminals are at distinct vertices and hence $\mathcal{P}_t \cap \mathcal{P}_{t'} = \emptyset$ for $t \neq t'$. For $v \in V$, a variable $x(v) \in [0, 1]$ indicates whether v is chosen in the solution or not. For $p \in \cup_t \mathcal{P}_t$ a variable $f(p) \in [0, 1]$ indicates whether p is used to connect a terminal to the root. We use $\ell(p)$ to denote $\sum_{v \in p} \ell(v)$. The LP assigns fractional capacities to vertices such that one unit of flow can be shipped from each terminal t to the root.

$$\text{LP-NSS} \quad \min \ \sum_{v \in V} c(v) \cdot x(v) + \sum_{t \in T} \delta(t) \sum_{p \in \mathcal{P}_t} \ell(p) \cdot f(p)$$

$$\begin{aligned} \text{subject to:} \quad \sum_{p \in P_t | v \in p} f(p) &\leq x(v) && v \in V, \ t \in T \\ \sum_{p \in \mathcal{P}_t} f(p) &\geq 1 && t \in T \\ x(v), f(p) &\geq 0 && v \in V, \ p \in \cup_t \mathcal{P}_t \end{aligned}$$

Randomized Algorithm for node-weighted single-sink buy-at-bulk:

1. Compute a subgraph Γ whose density is no more than the best density spider.
2. For every terminal $t \in \Gamma$, choose t to be a center with probability $p = \delta(t)/\sum_{t' \in T(\Gamma)} \delta(t')$. Set the demand of t to be equal to $\sum_{t' \in T(\Gamma)} \delta(t')$ and remove every other terminal $t \in T(\Gamma)$ from the terminal sets. For every non-center terminal in Γ, connect it to the root via t.
3. Continue recursively (i.e. goto Step 1)

Figure 1: A randomized algorithm for NSS-BB

Let OPT* be the optimum solution to the above LP and OPT* be its value. At every iteration i of the algorithm we will find a low density subgraph Γ_i. We show that we can find a subgraph Γ_i whose density is no more than the density of best density spider in polynomial time, where the density of a spider is its total cost over the number of terminals in the union of its leaves and its center. The total cost of a spider $\mathcal{S}$ with center s is:

$$c(s) + \sum_{t \in T(\mathcal{S})} (c(p_t) - c(s) + \delta(t) \cdot \ell(p_t)),$$

where $T(\mathcal{S})$ is the set of terminals in $\mathcal{S}$, and for every $t \in T(\mathcal{S})$, p_t is the path between t and s with $c(p_t)$ and $\ell(p_t)$ being the sum of the costs and lengths of the nodes on this path, respectively. Then we prove that the density of the best density spider is no more than OPT* over the total number of terminals h. Once we have a good density subgraph Γ_i (whose density is no more than the best density spider), we randomly pick one of the terminals t in Γ_i as a center proportional to the demand of that terminal to the total demands of the terminals in Γ_i and "route" all the demands of other terminals in Γ_i to t and remove them from the set of terminals. This is a modification of an idea of [27]. We will later show how to do this step deterministically. Note that if $r \in \Gamma_i$, because of our technical assumption about $\delta(r)$, r will be the center. We prove that the cost of the LP solution on this new modified instance (with fewer terminals) is at most OPT*. We repeat this process until all the demands are routed to root r. Finally, a set-cover type density analysis shows that the cost of the solution obtained is at most an $O(\log k)$ factor away from the optimum LP solution. For the ease of exposition, first we present a randomized version of the algorithm (see Figure 1).

Next lemma shows how to perform Step 1 of the algorithm and upper bounds the cost of the subgraph we find.

Lemma 4.3 *Given an instance of NSS-BB we can find in polynomial time a subgraph Γ in Step 1 whose density is no more than the density of best density spider.*

Proof. We run the following for every node $v \in V$ as being the center s of a spider. For every terminal $t \in T$, we compute its shortest path to the center s where the weight function for every vertex v (other than the center) on the path to compute the weight of the path is $c(v) + \delta(t) \cdot \ell(v)$. Now order the terminals in non-decreasing order of their shortest paths to the center. Without loss of generality assume that terminal t_i has the i'th shortest distance which is d_i. For every $1 \leq j \leq h$, we take the first j terminals (according to the above ordering) as the terminals in the best density spider with j terminals and its density is $\left(c(s) + \sum_{i=1}^{j} d_i\right)/j$. We pick the index j which minimizes this density and we return the subgraph Γ induced by the vertices of the union of these paths. It is easy to see that the density of this graph is no more than the density of best density spider.

□

Let I_i be the instance at the beginning of the ith iteration of the algorithm; we use index i to refer to the values of the variables in iteration i. So in iteration i, T_i will be the set of terminals, $h_i = |T_i|$, OPT$_i^*$ be the value of the optimum LP for instance I_i, Γ_i will be the best density subgraph found in Step 1, and $\mathcal{S}_i$ will be the best density spider.

The following lemma is the key lemma in the analysis of our algorithm (the proof of this lemma is relatively long and we defer it to Appendix A).

Lemma 4.4 *For every iteration $i \geq 1$, the density of the best density spider is no more than OPT_i^*/h_i.*

Lemma 4.5 *For every iteration $i \geq 1$: $\mathbf{E}(\text{OPT}_{i+1}^*) \leq \text{OPT}_i^*$.*

Proof. Let x_i^*, f_i^* be an optimal feasible solution to the instance I_i. Since in instance I_{i+1} only the value of demands have changed, x_i^*, f_i^* is also a feasible solution to I_{i+1}. We show that the expected cost of this solution on I_{i+1} is the same as OPT_i^*. To do so, for every terminal $t \in T_i$ let $\alpha(t) = \sum_{p \in \mathcal{P}_t} \ell(p) \cdot f_i^*(p)$. By this definition $\text{OPT}_i^* = \sum_{v \in V} c(v) \cdot x_i^*(v) + \sum_{t \in T_i} \delta(t) \cdot \alpha(t)$. For every terminal $t \notin \Gamma_i$, its contribution to the total cost remains unchanged. On the other hand, the expected contribution of the terminals of Γ_i in the I_{i+1} is exactly $\sum_{t \in \Gamma_i} \delta(t) \cdot \alpha(t)$. Since the x values have not changed, therefore, x_i^*, f_i^* is a feasible solution for I_{i+1} with expected value at most OPT_i^*. □

Proof of Theorem 4.1. Since at each iteration we aggregate some of the demands into one terminal, at the end we have a solution which routes all the demands to the root. Furthermore, the cost of routing the demands of the terminals of Γ_i to the center node selected is no more than twice the cost of Γ_i; therefore, using Lemmas 4.3 and 4.4, the cost added to the solution at each iteration i is at most $2|T(\Gamma_i)| \cdot \text{OPT}_i^*/h_i$. Using linearity of expectation and Lemma 4.5, the expected total cost of the solution is at most $\sum_{i \geq 1} 2|T(\Gamma_i)| \cdot \frac{\text{OPT}_i^*}{h_i} = \text{OPT}^* \sum_{i \geq 1} \frac{2|T(\Gamma_i)|}{h_i} = O(\log h).\text{OPT}^*$.

We can make the decision of selecting the center in Step 2 of the algorithm deterministic using the method of conditional probabilities. Once we have found the best density graph Γ_i, we consider each terminal $t \in T(\Gamma_i)$ as being a potential center. Then using the solution to the current LP we compute the cost of the modified LP which is obtained by assuming t being the center for re-routing in Γ_i (this can be easily done using the arguments given in the proof of Lemma 4.5). We choose the terminal $t \in T(\Gamma_i)$ which minimizes the LP cost as the center for this iteration. □

5 Node-Weighted Non-Uniform Multicommodity Buy-at-Bulk

Our main result of this section is as follows.

Theorem 5.1 *There is a polynomial time algorithm for NMC-BB with an $O(\min\{\log^3 h \log D, \log^5 h \log\log h\})$ approximation ratio, where h is the number of pairs and D is the sum of the demands of all pairs in $\mathcal{T}$.*

More specifically, our algorithm achieves an $O(\gamma(h^2)\log^3 h)$-approximation where $\gamma(n)$ is the worst case upper bound on the distortion in embedding a finite metric induced by a n vertex weighted undirected graph into a probability distribution over its spanning trees. It is known that $\gamma(n) = O(\log^2 n \log\log n)$ [11] and that $\gamma(n) = \Omega(\log n)$ [2]. The proof of Theorem 5.1 borrows ideas from [7] on edge-weighted multicommodity buy-at-bulk network (MC-BB) design. A main ingredient to obtain an $O(\log^5 h \log\log h)$-approximation ratio is to use the integrality gap obtained in Theorem 4.1. We also show that using a greedy algorithm similar to the one in [7] for MC-BB and also similar to the algorithm of Theorem 3.1 we can obtain an $O(\log^3 h \log D)$-approximation for NMC-BB. For this we develop a polylogarithmic approximation algorithm for a variation of the NSS-BB problem. This latter result is inspired by the work of Klein and Ravi [26] and our earlier work [22].

Overview of the algorithms: The general structure of our algorithms is similar to those in [7] for the MC-BB and follow a greedy scheme in an iterative fashion. In each iteration we find a partial solution that connects a subset of the pairs that remain at the beginning of the iteration. The connected pairs are then removed. The *density* of the partial solution is the ratio of the total cost of the partial solution to the number of pairs in the solution. We prove that the density of the partial solution computed at every iteration is a polylogarithmic factor away from the density of the optimum solution. As in [7], a key ingredient in our proof is to show the *existence* of a partial solution with a very restricted structure, called *junction-tree*. Given a subset A of the pairs, a junction tree for A rooted at r is a tree T containing the end points of all pairs in A such that the unique path connecting every pair of A goes via r. The cost of the junction-tree T is

$$\sum_{v \in V(T)} c_v + \sum_{s_i t_i \in A} \delta_i \cdot (\ell_T(r, s_i) + \ell_T(r, t_i)).$$

In other words, the pairs in A connect via the junction r. Note that if the set A and r are known, a junction-tree is essentially an instance of the single-sink problem NSS-BB. We prove that given an instance of NMC-BB there is always a low density partial solution that is a junction-tree. We give two different proofs; one achieves a better bound (by a logarithmic factor) for the uniform demand case while the other achieves a bound independent of D for the general case. The problem of finding a low density junction-tree is closely related to the density variation of NSS-BB, called den-NSS-BB in which we want to find a solution with minimum density i.e. the ratio of total cost over the number of terminals spanned (v.s. the total cost as in SS-BB). We present two different methods to compute a low density junction tree. For arbitrary demands we use an LP relaxation to solve the problem approximately. In particular we use Theorem 4.1 and obtain an $O(\log^2 h)$-approximation for den-NSS-BB and by a slight modification a similar ratio for finding the best density junction-tree. For the case that D is polynomial in h, we present a *greedy* algorithm, that is simple and efficient to implement. Putting together these ingredients give us the poly-logarithmic approximation for NMC-BB.

5.1 Two Junction Tree Lemmas

In this section we present two lemmas about the existence of junction trees. One works for arbitrary demand functions and gives an upper bound of $O(\gamma(h^2) \cdot \frac{\text{OPT}}{h})$. The other one (using a different proof technique) gives a better bound for the case that the sum of demands D is polynomial in h.

Lemma 5.2 (Junction tree lemma for arbitrary D) *Given an instance of NMC-BB on h pairs there exists a junction-tree of density $O(\gamma(h^2) \cdot \frac{\text{OPT}}{h})$.*

The proof of this lemma follows the same steps as Lemma 3.1 in [7]. In particular, we will need the following two lemmas. We only give the sketch of the proof of the second one as it is slightly different from that in [7].

Lemma 5.3 *Given an instance of NMC-BB on $G = (V, E)$ there is an optimum solution $G' = (V', E')$ such that the number of vertices in G' of degree more than 2 is at most $\min(n, h^2)$.*

Lemma 5.4 *Given an instance of NMC-BB on $G = (V, E)$ there is an $O(\gamma(h^2))$-approximate solution $G' = (V', E')$ such that G' is a forest.*

Proof Sketch. Consider an optimum solution $G' = (V', E')$ to NMC-BB. Without loss of generality we assume that G' is connected (as we can do the following to each of its connected component) and all vertices of V' are in our solution. The cost of the solution is $\sum_{v \in V'} c(v) + \sum_{i=1}^{h} \delta_i \cdot \ell_{G'}(s_i, t_i)$. We obtain a new graph G'' from G' on the same vertex and edge set except that the edges (instead of the vertices) have lengths. For each edge $e = uv \in G''$ assign a length $\ell(e) = (\ell(u) + \ell(v))/2$; now drop the costs and lengths on the vertices. An observation that will be used soon is that for any path $p \subset G''$, the length of the path in G'' is within factor two of the length of the corresponding path in G'.

From the definition of $\gamma(n)$, there is a probability distribution over the spanning trees of G'' with the following property: for any pair of vertices uv, their expected distance in a tree chosen from the distribution is at most $\gamma(n)$ times their distance in G''. Using linearity of expectation and the observation stated above, this implies the existence of a tree T in G'' such that the sum of the distances of the pairs times their demands in T is no more than $\gamma(n)$ times the sum of the distances of the pairs times their demands in G', i.e.

$$\sum_i \delta_i \cdot \ell_T(s_i, t_i) \leq \gamma(n) \cdot \sum_i \delta_i \cdot \ell_{G'}(s_i, t_i).$$

Note that on the LHS, ℓ_T is w.r.t. edge lengths v.s. on the RHS, $\ell_{G'}$ is w.r.t. vertex lengths. Since T contains all the vertices of G'', and G'' and G' have the same vertex set, the fixed cost of vertices in T is at most the fixed cost of vertices in our initial solution. Since the edges of T are a subset of those in G'' and thus G', the edge-length of a path in T is within a factor 2 of the vertex-length of the corresponding path in G'. Therefore, the tree in G' corresponding to T is an $O(\gamma(n))$-approximation to the optimal solution. We can use Lemma 5.3 to improve the bound to $\gamma(h^2)$ when h is small compared to n. □

The proof of the following lemma follows similar steps as the proof of Lemma 3.4 in [7]. We skip the details.

Lemma 5.5 (Junction tree lemma for polynomially bounded D) *Given an instance of NMC-BB with unit demands there is a junction-tree of density $O(\log h \cdot \frac{\mathrm{OPT}}{h})$. For the general case with total demand D, there exists a junction-tree of density $O(\log h \cdot \frac{\mathrm{OPT}}{D})$.*

5.2 Approximation Algorithms for Min-density Junction Tree

In this subsection we give an $O(\log^2 h)$-approximation algorithm for den-NSS-BB and min-density junction tree. Consider the following LP relaxation of den-NSS-BB. For each terminal t_i, we have an additional variable y_i that indicates whether t_i is chosen in the solution or note. We have normalized the sum $\sum_t y_t$ to 1.

$$\text{LP-NSSD} \quad \min \quad \sum_{v\in V} c(v)\cdot x(v) + \sum_{t\in T}\delta(t)\sum_{p\in\mathcal{P}_t}\ell(p)\cdot f(p)$$

$$\begin{array}{rrcll} \text{subject to:} & \sum_{t\in T} y_t & = & 1 & \\ & \sum_{p\in P_t | v\in p} f(p) & \leq & x(v) & v\in V,\ t\in T \\ & \sum_{p\in\mathcal{P}_t} f(p) & \geq & y_t & t\in T \\ & x(v), f(p), y_t & \geq & 0 & v\in V,\ p\in\cup_t\mathcal{P}_t \end{array}$$

Theorem 5.6 *There is an $O(\log^2 h)$-approximation for den-NSS-BB.*

Corollary 5.7 *There is an $O(\log^2 h)$-approximation for computing min-density junction tree.*

The reader is referred to Appendix A to see proofs of Theorem 5.6 and Corollary 5.7.

5.3 A Greedy Approximation Algorithms for Min-density Junction Tree

Here we describe the overview of an algorithm for NMC-BB with ratio $O(\log^3 h \cdot \log D)$. The algorithm is essentially the same as the greedy algorithm for MC-BB in [7] and follows similar steps to the algorithm of Theorem 3.1. In other words, it tries to find a partial solution with good density at every iteration. We describe briefly the general idea of the algorithm for MC-BB (from [7]) and the differences with the one for NMC-BB. The main ingredient in the greedy algorithm for MC-BB is an approximation algorithm for the shallow-light trees described here.

Shallow-light k-Steiner Tree (KSLT): The instance to shallow-light k-Steiner problem is a graph $G(V,E)$, with edge-weight function $c: E\to\mathcal{R}^+$ and edge-length function $\ell: E\to\mathcal{R}^+$, a collection T of terminals containing a root s, a number k, and a diameter bound L. The goal is to find an s-rooted k-Steiner tree that has ℓ-diameter at most L, and among all such subtrees, find the one with minimum c-cost. A (ρ_1,ρ_2)-approximation algorithm for the shallow-light k-Steiner problem finds an s-rooted k-Steiner tree with diameter at most $\rho_1\cdot L$ and cost at most $\rho_2\cdot B$ with B being the optimum cost for a k-Steiner tree of diameter L. The algorithm in [7], uses the following result from [22] for the *edge-weighted* version of shallow-light trees:

Theorem 5.8 *[22] There is an $(O(\log h), O(\log^3 h))$-approximation algorithm for the edge-weighted shallow-light k-Steiner tree problem which finds a $k/8$-Steiner tree.*

The algorithm for MC-BB follows steps similar to those of algorithm of Theorem 3.1. The main procedure which tries to find a good density partial solution has a sources-phase and sinks-phase and at each phase it finds best-density trees rooted at a node s. For that purpose, it uses the algorithm of Theorem 5.8 (instead of D-NWST). The analysis is similar to that of Theorem 3.1 but somewhat more involved (because here we have two weight functions c and ℓ); we skip the details of the analysis from [7].

Our greedy algorithm for the NMC-BB follows the same paradigm. For that we need a node-weighted version of Theorem 5.8. We define the node-weighted shallow-light Steiner trees similarly:

Node-weighted shallow-light k-Steiner tree (NKSLT): we have a graph $G(V,E)$, with node cost function $c: V\to\mathcal{R}^+$ and length function $\ell: V\to\mathcal{R}^+$, a collection T of terminals containing a root s, a number k, and a diameter bound L. The goal is to find an s-rooted k-Steiner tree that has ℓ-diameter at most L, and among all such subtrees, find the one with minimum c-cost.

Lemma 5.9 *There is polynomial-time algorithm A s.t. given an instance of NKSLT, finds a $k/8$-Steiner tree with diameter at most $O(\log h \cdot L)$ and cost at most $O(\log^3 h \cdot \text{OPT})$ where OPT the cost of an optimum k-Steiner solution with diameter bound L.*

Using this lemma, an algorithm similar to the one for MC-BB gives an $O(\log^3 h \log D)$-approximation for NMC-BB. So what is left is to prove Lemma 5.9. This algorithm borrows ideas from the algorithm of [22] for (edge-weighted) shallow-light k-Steiner trees (Theorem 5.8) and [26] for node-weighted Steiner tree. Here we briefly describe the similarities and differences. The algorithm for Theorem 5.8 is a greedy algorithm that starts from every terminal as a single-component. At every iteration it tries to connect two components by a "cheap" path. Once a path is found the two components are merged into one. We continue until we have a component with at least $k/8$ terminals. The exact definition of a cheap path is such that we can charge the cost of the path to the nodes in the two components merged and this cost should be at most a polylogarithmic factor of the optimum density (there are some technical details that we omit here). The algorithm for Lemma 5.9 has a similar structure. The main difference is that at each iteration, instead of finding a cheap path that connect two terminals (at good density) we try to find a best density spider. The algorithm for finding the best density spider is the same as the one for Lemma 4.3. Once we have found a good density spider (compared to the density of the optimum) we merge the components it spans. We continue this until there are at least $k/8$ terminals in one component.

References

[1] A. AGRAWAL, P. KLEIN, AND R. RAVI, *When trees collide: an approximation algorithm for the generalized Steiner problem on networks*, in STOC, 1991, pp. 134–144.

[2] N. ALON, R. M. KARP, D. PELEG, AND D. B. WEST, *A graph-theoretic game and its application to the k-server problem*, SIAM J. Computing, 24 (1995), pp. 78–100.

[3] M. ANDREWS, *Hardness of buy-at-bulk network design.*, in Proceedings of the 45th Symposium on Foundations of Computer Science (FOCS '04), 2004, pp. 115–124.

[4] M. ANDREWS AND L. ZHANG, *Approximation algorithms for access network design*, Algorithmica, 34 (2002), pp. 197–215.

[5] B. AWERBUCH AND Y. AZAR, *Buy-at-bulk network design*, in Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97), IEEE Computer Society, 1997, p. 542.

[6] M. CHARIKAR AND A. KARAGIOZOVA, *On non-uniform multicommodity buy-at-bulk network design*, in Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (STOC '05), New York, NY, USA, 2005, ACM Press, pp. 176–182.

[7] C. CHEKURI, M. HAJIAGHAYI, G. KORTSARZ, AND M. SALAVATIPOUR, *Approximation algorithms for non-uniform buy-at-bulk network design problems*, in Proceedings of 47th Symposium on Foundations of Computer Science (FOCS), Berkeley, CA, USA, 2006. to appear.

[8] C. CHEKURI, S. KHANNA, AND J. NAOR, *A deterministic algorithm for the cost-distance problem*, in Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms (SODA '01), Philadelphia, PA, USA, 2001, Society for Industrial and Applied Mathematics, pp. 232–233.

[9] C. CHEKURI, S. KHANNA, AND F. B. SHEPHERD, *Multicommodity flow, well-linked terminals, and routing problems*, in Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (STOC '05), New York, NY, USA, 2005, ACM Press, pp. 183–192.

[10] J. CHUZHOY, A. GUPTA, J. S. NAOR, AND A. SINHA, *On the approximability of some network design problems*, in Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '05), Philadelphia, PA, USA, 2005, Society for Industrial and Applied Mathematics, pp. 943–951.

[11] M. ELKIN, Y. EMEK, D. A. SPIELMAN, AND S.-H. TENG, *Lower-stretch spanning trees*, in Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (STOC '05), New York, NY, USA, 2005, ACM Press, pp. 494–503.

[12] J. FAKCHAROENPHOL, S. RAO, AND K. TALWAR, *A tight bound on approximating arbitrary metrics by tree metrics*, in Proceedings of the thirty-fifth annual ACM symposium on Theory of computing (STOC '03), New York, NY, USA, 2003, ACM Press, pp. 448–455.

[13] U. FEIGE, M. HAJIAGHAYI, AND J. R. LEE, *Improved approximation algorithms for minimum-weight vertex separators*, in Proceedings of the thirty-seventh annual ACM Symposium on Theory of Computing (STOC '05), New York, NY, USA, 2005, ACM Press, pp. 563–572.

[14] L. FLEISCHER, K. JAIN, AND D. P. WILLIAMSON, *An iterative rounding 2-approximation algorithm for the element connectivity problem*, in FOCS, 2001, pp. 339–347.

[15] A. FRANK, *Matroids and submodular functions*, in Annotated Bibliographies in Combinatorial Optimization, (M. Dell Amico, F. Maffioli, S. Martello, eds.), 1997, pp. 65–80.

[16] M. X. GOEMANS AND D. P. WILLIAMSON, *A general approximation technique for constrained forest problems*, SIAM J. Comput., 24 (1995), pp. 296–317.

[17] S. GUHA, A. MEYERSON, AND K. MUNAGALA, *A constant factor approximation for the single sink edge installation problems*, in Proceedings of the thirty-third annual ACM symposium on Theory of computing (STOC '01), New York, NY, USA, 2001, ACM Press, pp. 383–388.

[18] S. GUHA, A. MOSS, J. NAOR, AND B. SCHIEBER, *Efficient recovery from power outage*, in STOC, 1999, pp. 574–582.

[19] A. GUPTA, A. KUMAR, M. PAL, AND T. ROUGHGARDEN, *Approximation via cost-sharing: a simple approximation algorithm for the multicommodity rent-or-buy problem*, in Proceedings of the 44rd Symposium on Foundations of Computer Science (FOCS '03), IEEE Computer Society, 2003, p. 606.

[20] A. GUPTA, A. KUMAR, AND T. ROUGHGARDEN, *Simpler and better approximation algorithms for network design*, in Proceedings of the thirty-fifth ACM symposium on Theory of computing (STOC '03), ACM Press, 2003, pp. 365–372.

[21] M. HAJIAGHAYI, R. D. KLEINBERG, T. LEIGHTON, AND H. RÄCKE, *Oblivious routing on node-capacitated and directed graphs*, in Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA '05), ACM Press, 2005, pp. 782–790.

[22] M. HAJIAGHAYI, G. KORTSARZ, AND M. R. SALAVATIPOUR, *Approximating buy-at-bulk and shallow-light k-steiner tree*, in Proceedings of 9th. International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX), 2006. to appear.

[23] D. HOCHBAUM, *Approximation algorithms for NP-hard problems*, PWS Publishing Company, 1997.

[24] K. JAIN, *A factor 2 approximation algorithm for the generalized steiner network problem*, Combinatorica, 21 (2001), pp. 39–60.

[25] D. S. JOHNSON, *Approximation algorithms for combinatorial problems*, J. Comput. System Sci., 9 (1974), pp. 256–278. Fifth Annual ACM Symposium on the Theory of Computing (Austin, Tex., 1973).

[26] M. V. MARATHE, R. RAVI, R. SUNDARAM, S. S. RAVI, D. J. ROSENKRANTZ, AND H. B. HUNT, III, *Bicriteria network design problems*, J. Algorithms, 28 (1998), pp. 142–171.

[27] A. MEYERSON, K. MUNAGALA, AND S. PLOTKIN, *Cost-distance: two metric network design*, in Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS '00), IEEE Computer Society, 2000, p. 624.

[28] A. MOSS AND Y. RABANI, *Approximation algorithms for constrained for constrained node weighted steiner tree problems*, in STOC, 2001, pp. 373–382.

[29] F. S. SALMAN, J. CHERIYAN, R. RAVI, AND S. SUBRAMANIAN, *Approximating the single-sink link-installation problem in network design*, SIAM J. on Optimization, 11 (2000), pp. 595–610.

[30] A. SCHRIJVER, *Combinatorial optimization*, Springer, 2003.

[31] K. TALWAR, *The single-sink buy-at-bulk lp has constant integrality gap*, in Proceedings of the 9th International IPCO Conference on Integer Programming and Combinatorial Optimization, London, UK, 2002, Springer-Verlag, pp. 475–486.

[32] V. VAZIRANI, *Approximation algorithms*, Springer, 1994.

Procedure Partial
Input: $\mathcal{T}'$ the set of remaining pairs to be connected
Output: A partial solution connecting at least one pair of $\mathcal{T}'$

Let $\mathcal{T}'' \leftarrow \mathcal{T}'$ and $h' = |\mathcal{T}'|$;

While $\mathcal{T}'' \neq \emptyset$ **Do**

Let s be an arbitrary source in $\mathcal{T}''$

$\text{LowDens} \leftarrow \text{True}$; $V_s \leftarrow \emptyset$

Repeat /* The sources-phase starts /*

Find the best density tree F_s rooted at s with sources being the terminals

if $c(F_s)/|T(F_s)| \leq 8\alpha.\text{OPT}/h'$ **then**

- Add all the vertices of F_s to V_s
- Contract all of F_s into s and set the cost of s to zero.

else $\text{LowDens} \leftarrow \text{False}$

Until $\text{LowDens} = \text{False}$

Let X be set of sources in V_s and Y be the sinks whose corresponding source is in X.

$V_t \leftarrow \emptyset$ and let $x = |X|$

Repeat /* The sinks-phase starts /*

Find the best density tree F_t rooted at s with the nodes in Y being the terminals.

Add all the vertices of F_t to V_t

Contract all of F_t into s and set the cost of s to zero.

Until (V_t has at least $\lceil x/32 \rceil$ sinks of the sources in X)

If density of $V_s \cup V_t$ is at most $384\alpha \cdot \text{OPT}/h'$ **then** return $V_s \cup V_t$

else discard from $\mathcal{T}''$ all the pairs whose sources are in X.

Figure 2: Procedure Partial that is called iteratively in the main algorithm for Theorem 3.1

A Omitted Proofs

Proof of Lemma 4.4. Consider some iteration $i \geq 1$ of the algorithm and let R_i be the ratio of the best density spider $\mathcal{S}_i$ (so the density of subgraph Γ_i is no more than R_i). For every terminal t we compute a ball of radius R_i with the center of t in the following way. For each vertex v we define the weight of v to be $c(v) + \delta(t) \cdot \ell(v)$. Now the distance of every vertex v from t is the sum of the weights of the nodes on the shortest path from v to t using this weight function (counting v too); we denote it by $\text{dist}(v, t)$. A ball of radius R_i around t contains some vertices fully and some vertices partially; for every vertex v and terminal t_j, $0 \leq \gamma_{t_j}(v) \leq 1$ is the fraction of *cost* of v and $0 \leq \sigma_{t_j}(v) \leq 1$ is the fraction of *length* of v that belongs to ball of t_j. We will maintain the property:

$$\forall v : \quad \sum_{t_j \in T} \gamma_{t_j}(v) \leq 1. \tag{4}$$

Also, whenever $\gamma_{t_j}(v) > 0$ we will have $\sigma_{t_j}(v) = 1$. All the nodes v with $\text{dist}(v, t_j) \leq R_i$ are fully contained in the ball of terminal t_j, denoted by B_{t_j}; so $\gamma_{t_j}(v) = \sigma_{t_j}(v) = 1$. First note that:

Claim A.1 *If a vertex v fully belongs to a ball B_{t_j} (i.e. $\gamma_{t_j}(v) = 1$) then it cannot fully belong to any other ball B_{t_k}.*

Proof. We prove this by way of contradiction. Assume that v fully belongs to two balls B_{t_j} and B_{t_k}. Therefore, if p_j and p_k are the paths from t_j and t_k to v respectively, then $\text{dist}(v, t_j) = \delta(t_j) \cdot \ell(p_j) + c(p_j) \leq R_i$ and

$\text{dist}(v, t_k) = \delta(t_k) \cdot \ell(p_k) + c(p_k) \leq R_i$. WLOG assume that $\delta(t_j) \leq \delta(t_k)$ and consider the spider which consist of the path between t_j and t_k with t_k being the center. The density of this spider is at most $\frac{1}{2}[\delta(t_j) \cdot (\ell(p_j) + \ell(p_k) - \ell(v)) + c(p_j) + c(p_k) - c(v)]$ (we have subtracted $\ell(v)$ and $c(v)$ because v appears in both paths). Since at least one of $c(v)$ or $\ell(v)$ is positive, this ratio is strictly smaller than R_i, contradicting the assumption that R_i is the ratio of the best density spider.

□

Now we describe how the vertices that do not fully belong to any ball can partially be part of a ball. For a terminal t_j and vertex v that does not fully belong to any ball, suppose that $\text{dist}(v, t) > R_i$ and let $p_{t_j,v}$ be the shortest path from t_j to v. If the last vertex on the path before v, call it u, has $\text{dist}(u, t) < R_i$ then we say ball B_{t_j} is within reach of v and v will belong to this ball fractionally. We define this fraction as follows. Assume that balls $B_{t_{a_1}}, \ldots, B_{t_{a_k}}$ (with centers $t_{a_1}, \ldots, t_{a_k}$) are the balls that are within reach of v. Let $d_{t_{a_j}}$ $(1 \leq j \leq k)$ be the distance from t_{a_j} to the last vertex (before v) on the shortest path from t_{a_j} to v; so $d_{t_{a_j}} < R_i$. Define $\rho_{t_{a_j}} = R_i - d_{t_{a_j}}$. In a sense, we could still continue on the shortest path from t_{a_j} to v for up to $\rho_{t_{a_j}}$ before the distance becomes R_i. If $\rho_{t_{a_j}} \geq \delta(t_{a_j}) \cdot \ell(v)$ then we define $\sigma_{t_{a_j}}(v) = 1$; otherwise $\sigma_{t_{a_j}}(v) = \rho_{t_{a_j}} / (\delta(t_{a_j}) \cdot \ell(v))$.

Claim A.2 *For every vertex v that is within reach of the balls $B_{t_{a_1}}, \ldots, B_{t_{a_k}}$:*

$$c(v) + \ell(v) \sum_{j=1}^{k} \sigma_{t_{a_j}}(v) \cdot \delta(t_{a_j}) \geq \sum_{j=1}^{k} \rho_{t_{a_j}}.$$

Proof. By way of contradiction assume not. First consider those terminals t_{a_j} for which $\sigma_{t_{a_j}}(v) < 1$. For these, $\sigma_{t_{a_j}}(v) \cdot \ell(v) \cdot \delta(t_{a_j}) = \rho_{t_{a_j}}$. So we can subtract them from both sides. What remains are those terminals t_{a_j} for which $\sigma_{t_{a_j}}(v) = 1$. For simplicity, let's assume that for all $t_{a_1}, \ldots, t_{a_k}$ we have $\sigma_{t_{a_j}} = 1$. Then the union of shortest paths from t_{a_j}'s to v forms a spider with center v and total cost at most $\left(\sum_{j=1}^{k} d_{t_{a_j}}\right) + \left(c(v) + \ell(v) \sum_{j=1}^{k} \delta(t_{a_j})\right) < \sum_{j=1}^{k} (d_{t_{a_j}} + \rho_{t_{a_j}}) = k \cdot R_i$. Thus there is a spider with density smaller than R_i, a contradiction. □

Now we are ready to define the fraction $\gamma_{t_{a_j}}(v)$ of (cost of) vertex v that belongs to ball $B_{t_{a_j}}$ (with center t_{a_j}). We define:

$$\gamma_{t_{a_j}}(v) = \frac{\rho_{t_{a_j}} - \sigma_{t_{a_j}}(v) \cdot \delta(t_{a_j}) \cdot \ell(v)}{c(v)}.$$

Note that by this definition, if $\sigma_{t_{a_j}}(v) < 1$ then $\gamma_{t_{a_j}}(v) = 0$ and if $\sigma_{t_{a_j}} = 1$ then $\gamma_{t_{a_j}}(v) \geq 0$. It is not hard to see that from Claims A.1 and A.2 it follows that $\sum_{j=1}^{k} \gamma_{t_{a_j}}(v) \leq 1$. So we maintain Inequality (4) as wanted. For consistency, if a vertex v does not belong (fully or partially) to any ball we set $\gamma_t(v) = 0$ for all terminals t. Finally we note that a vertex that fully belongs to any ball cannot fully or even partially belong to any other ball.

Now we prove $\text{OPT}_i^* \geq h_i \cdot R_i$. From this, the lemma follows immediately. To prove this, we show the existence of a dual feasible solution with value at least $h_i \cdot R_i$. Since the value of the dual solution is a lower bound for OPT_i^* the statement follows. Below is the dual program to LP-NSS:

$$\begin{array}{lrcll} \text{DP-NSS} & \max & \sum_{t \in T} y(t) & & \\ \text{subject to:} & \sum_{t \in T} z_t(v) & \leq & c(v) & v \in V \\ & y(t) - \sum_{v \in p} z_t(v) & \leq & \delta(t) \cdot \ell(p) & p \in \mathcal{P}_t, \;\; t \in T \\ & y(t), z_t(v) & \geq & 0 & v \in V, t \in T \end{array}$$

Consider the following solution to DP-NSS: set $y(t) = R_i$ for all terminals t and for every vertex v set $z_t(v) = \gamma_t(v) \cdot c(v)$. We claim that this is a feasible solution to DP-NSS. Using Equation (4), constraint $\sum_{t \in T} z_t(v) \leq c(v)$ is never violated. Now we consider the other constraints. Let $t \in T$ be an arbitrary terminal and $p \in \mathcal{P}_t$ an arbitrary path.

- *Case 1:* if all the vertices on path p either fully belong to B_t or do not belong to B_t at all then for the last vertex u on p that fully belongs to B_t we must have $\mathrm{dist}(u,t) = R_i$. Otherwise either the next vertex on the path partially belongs to B_t (which contradicts our assumption) or the next vertex is r, in which case path p has distance strictly smaller than R_i and so forms a spider with density smaller than R_i. Thus $\delta(t) \cdot \ell(p) + \sum_{v \in p} z_t(v) = \delta(t) \sum_{v \in p, \gamma_t(v)=1} \sigma_t(v) \cdot \ell(v) + \sum v \in p z_t(v) \geq R_i = y(t)$, or equivalently $y(t) - \sum_{v \in p} z_t(v) \geq \delta(t) \cdot \ell(p)$.

- *case 2:* Let u be the first vertex on path p (from t to r) with $0 < \gamma_t(u) < 1$. Note that all the vertices before u on p fully belong to B_t. Let $d_t = \sum_{v \in p, \gamma_t(v)=1} (c(v) + \sigma_t(v) \cdot \delta(t) \cdot \ell(v))$. We should point out that for all v's considered in this sum, $\sigma_t(v) = 1$ (because $\gamma_t(v) = 1$), and that $d_t < R_i$ (because u partially belongs to the ball of t). Let $p - u$ denote the path p with vertex u removed. Since $\gamma_t(u) \cdot c(u) + \sigma_t(u) \cdot \delta(t) \cdot \ell(u) = \rho_t$ (by definition of $\gamma_t(u)$) and $d_t + \rho_t = R_i$ we have:

$$\begin{aligned}
\delta(t) \cdot \ell(p) + \sum_{v \in p} z_t(v) &= \left(\delta(t) \cdot \ell(p-u) + \sum_{v \in p-u} z_t(v) \right) + (\delta(t) \cdot \ell(u) + z_t(u)) \\
&\geq d_t + \sigma_t(u) \cdot \delta(t) \cdot \ell(u) + \gamma_t(u) \cdot c(u) \\
&= d_t + \rho_t \\
&= R_i = y(t).
\end{aligned}$$

  Or equivalently $y(t) - \sum_{u \in p} z_t(u) \leq \delta(t) \cdot \ell(p)$.

Hence, none of the constraints are violated and so there is a feasible solution to DP-NSS with value $h_i \cdot R_i$. This is also a lower bound for $\textsc{opt}_i^*$. □

Proof of Theorem 5.6. The proof is similar to that of Theorem 4.2 in [7]. The main difference is that here we use Theorem 4.1. Consider an optimum solution to LP-NSSD. We obtain disjoint subsets of the terminals $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_p$ as follows. Let $y_{\max} = \max_t y_t$. For $0 \leq a \leq 2\lceil \log h \rceil$, let $\mathcal{T}_a = \{t_j \mid y_{\max}/2^{a+1} < y_{t_j} \leq y_{\max}/2^a\}$. Thus $p = 1 + 2\lceil \log h \rceil = O(\log h)$. It is easy to see that there is an index b such that $\sum_{t_j \in \mathcal{T}_b} y_{t_j} = \Omega(1/\log h)$. From this we also have that $2^b/|\mathcal{T}_b| = O(\log h)$. We now solve an NSS-BB instance on $\mathcal{T}_b$. We claim that the resulting solution is an $O(\log^2 h)$-approximation to den-NSS-BB. To prove this, let α be the value of the optimum solution to LP-NSSD on the given instance. Note that if we scale up, by a factor of $2^{b+1}/y_{\max}$, the given optimum solution to LP-NSSD we obtain a feasible solution to LP-NSS on the terminal set $\mathcal{T}_b$. The cost of this scaled solution to LP-NSS is $2^{b+1}\alpha$. Since the integrality gap of LP-NSS is $O(\log h)$ (by Theorem 4.1), we obtain an integral solution that connects each terminal in $\mathcal{T}_b$ to the root such that cost of the solution is $O(\log h) \cdot 2^{b+1}\alpha$. The density of this solution is therefore $O(\log h) \cdot 2^{b+1}\alpha/|\mathcal{T}_b|$ which is $O(\log^2 h)\alpha$. Thus the integrality gap of LP-NSSD is $O(\log^2 h)$ yielding the desired approximation. □

Proof of Corollary 5.7. Given an instance of NMC-BB, we consider each source or sink as a terminal. Also, for every pair s_i, t_i we add the following set of constraints to the LP-NSSD: $y_{s_i} = y_{t_i}$. This ensures that either we include both of s_i and t_i in the tree or none of them. The rounding scheme in the proof of Theorem 5.6 extends to this LP and so we get an $O(\log^2 h)$-approximation for the min-density junction tree problem. □