Approximation Algorithms for Network Design with Metric Costs

Joseph Cheriyan*
Department of Combinatorics and Optimization
University of Waterloo
jcheriyan@uwaterloo.ca

Adrian Vetta†
Department of Mathematics and Statistics, and
School of Computer Science
McGill University
vetta@math.mcgill.ca

ABSTRACT

We study undirected networks with edge costs that satisfy the triangle inequality. Let n denote the number of nodes. We present an $O(1)$-approximation algorithm for a generalization of the metric-cost subset k-node-connectivity problem. Our approximation guarantee is proved via lower bounds that apply to the simple edge-connectivity version of the problem, where the requirements are for edge-disjoint paths rather than for openly node-disjoint paths. A corollary is that, for metric costs and for each $k = 1, 2, \ldots, n-1$, there exists a k-node connected graph whose cost is within a factor of 24 of the cost of any simple k-edge connected graph. This resolves an open question in the area. Based on our $O(1)$-approximation algorithm, we present an $O(\log r_{\max})$-approximation algorithm for the node-connectivity survivable network design problem where $r_{\max}$ denotes the maximum requirement over all pairs of nodes. Our results contrast with the case of edge costs of zero or one, where Kortsarz et al. [20] recently proved, assuming NP$\not\subseteq$ quasi-P, a hardness-of-approximation lower bound of $2^{\log^{1-\epsilon} n}$ for the subset k-node-connectivity problem, where ϵ denotes a small positive number.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms

Algorithms, Theory

Keywords

Graph connectivity, approximation algorithms, metric costs

*Supported in part by NSERC grant OGP0138432.

†Supported in part by NSERC grant 28833−04 and FQRNT grant NC−98649.

STOC'05, May 22-24, 2005, Baltimore, Maryland, USA.

1. INTRODUCTION

A basic problem in network design is to find a minimum-cost sub-network H of a given network G such that H satisfies some prespecified connectivity requirements. Fundamental examples include the *minimum spanning tree* (MST) problem and the *traveling salesman problem* (TSP). By a *network* we mean an undirected graph together with non-negative costs for the edges, and we use n to denote the number of nodes. Our focus is on networks where the edge costs are metric; that is, the edge costs satisfy the triangle inequalities. This special case is significant from both theoretical and practical viewpoints; metric costs arise in many applications of network design, and perhaps in most of the obvious ones, such as the design of telecommunication networks. Our goal is to design and analyse approximation algorithms for some key problems in network design. Moreover, we resolve a long-standing conjecture on metric graphs, where by a *metric graph* we mean a complete graph K_n together with edge-costs that satisfy the triangle inequalities.

We attack the metric-cost *node-connectivity survivable network design problem* (NC-SNDP). In this problem, we are given a metric graph, as well as a connectivity requirement $r_{i,j}$ between every pair of nodes i and j. Let $r_{\max}$ denote $\max_{i,j \in V} r_{i,j}$. The goal is to find a minimum-cost subgraph H that satisfies these requirements, that is, H should have $r_{i,j}$ openly node-disjoint paths between every pair of nodes i and j. There are two well-known special cases of NC-SNDP. The first is the *subset k-node-connectivity problem*, where we are given a set of terminal nodes $T \subseteq V$ and $r_{i,j} = k$ precisely if both i and j are in T, otherwise $r_{i,j} = 0$. The second is the classical *k-node connected spanning subgraph problem* (k-NCSS) where $r_{i,j} = k$ for every pair of nodes; this is the special case of the subset k-node-connectivity problem with $T = V$. We also study a new special case of NC-SNDP that we call the *subset $[k,\ 1.5k]$-node-connectivity* problem: given a set of terminal nodes $T \subseteq V$ and an (integer) requirement r_i for each node $i \in T$, where $1 \le k \le r_i \le 1.5k$, the goal is to find a minimum-cost subgraph that has $\min(r_i, r_j)$ openly node-disjoint i,j-paths for every pair of nodes $i, j \in T$. (Thus the subset k-node-connectivity problem is the special case where $r_i = k,\ \forall i \in T$.) See Section 4 for more discussion.

Most network design problems stay NP-hard and APX-hard even assuming metric costs. This remains true even for small connectivity requirements; for example, Bern & Plassmann [3] showed that the Steiner tree problem (the classical special case of the subset k-node-connectivity problem with $k = 1$) is APX-hard even with edge costs of 1 and 2. Over the past decade, there has been significant research on approximation algorithms for network design, and there have been some notable successes in the design of networks that satisfy various types of "edge connectivity" requirements, e.g., Goemans & Williamson [16], and Jain [17], but from the perspective of approximation algorithms, the design of networks subject to "node connectivity" requirements is a murky area. For example, Kortsarz, Krauthgamer & Lee [20] recently proved a hardness-of-approximation lower bound of $2^{\log^{1-\epsilon} n}$ for the subset k-node connectivity problem in graphs with zero-one edge costs, provided that NP$\not\subseteq$ DTIME($n^{polylog(n)}$), where, ϵ denotes a small positive real number. (We give a detailed discussion on previous work in the area after stating our results.)

We present a 24-approximation algorithm for the metric cost subset k-node-connectivity problem, and then we generalize this to get an $O(1)$-approximation algorithm for the metric cost subset $[k, 1.5k]$-node connectivity problem. Modulo P$\neq$NP and up to constant factors, these are the best possible results. These algorithms are deterministic and combinatorial; they do not use linear programming relaxations. Based on this, we present an $O(\log r_{\max})$ approximation algorithm for the metric-cost NC-SNDP. The algorithm for NC-SNDP is based on a linear programming relaxation. Also, it uses a 2-approximation algorithm of Goemans & Williamson [16] (see also Agrawal et al. [1]) for the generalized Steiner tree problem. Moreover, we resolve the following long-standing conjecture: In a metric graph and for each $k = 1, 2, \ldots, n-1$, the minimum cost of a k-node connected spanning subgraph is within a constant factor of the minimum cost of a simple k-edge connected spanning subgraph. Thus, for metric graphs, the requirements of k-node-connectivity and simple k-edge-connectivity are equivalent for the objective function, up to constant factors. A similar result holds for requirements of subset $[k,\ 1.5k]$-node connectivity versus subset simple $[k,\ 1.5k]$-edge connectivity.

We apply two lower bounds on the optimal value of the subset $[k,\ 1.5k]$-connectivity problem. We may assume (without loss of generality) that there exist at least two terminals with the maximum requirement. Hence, every solution subgraph has at least r_i edges incident to each terminal i, because there is another terminal j with $r_j \geq r_i$, so the solution subgraph must have r_i openly node-disjoint i, j-paths. Our first lower bound comes from the the minimum cost of a subgraph that has degree $\geq r_i$ for every terminal i. Our second lower bound comes from the cost of a minimum spanning tree of the subgraph induced by the terminals. For any node i, we use σ_i or $\sigma(i)$ to denote the sum of the costs of the r_i cheapest edges incident to i in the complete graph, and for any set of nodes S, we use $\sigma(S)$ to denote $\sum_{i\in S}\sigma_i$. We use the abbreviations MST for minimum-cost spanning tree, and TSP for the traveling salesman problem. Let $mst(T)$ denote the cost of an MST of the subgraph induced by T. Our lower bounds are:

(i) $\frac{1}{2}\,\sigma(T)$, and

(ii) $\frac{k}{2}\ mst(T)$.

Note that these lower bounds apply also to the simple edge-connectivity version of the subset $[k,\ 1.5k]$-connectivity problem, where the requirements are for $\min(r_i, r_j)$ edge-disjoint paths between every pair of nodes $i, j \in T$; note that multi-edges are not allowed in the solution subgraph. See Section 2 for more details. Throughout, we use OPT to denote the cost of an optimal solution. Next, we state our main results formally.

THEOREM 1. *There is a polynomial-time algorithm for computing a solution to the metric-cost subset k-node connectivity problem of cost*

$$\leq 10\sigma(T) + 4(\frac{k}{2})\ mst(T) \leq 24\text{OPT}.$$

Consider k-NCSS, the special case of the subset k-node connectivity problem in which the terminal set T is V. Let k-ECSS be the problem of finding a minimum-cost *simple* k-edge connected spanning subgraph. Then our two lower bounds apply for both k-NCSS and k-ECSS. This gives the next result.

COROLLARY 2. *In a network with metric costs, there is a k-node connected spanning subgraph whose cost is at most 24 times the minimum cost of a simple k-edge connected spanning subgraph.*

Remarks: For metric graphs, it is well known that there exists a 2-node connected graph of cost $\leq$ the cost of any 2-edge connected graph, but this does not hold for $k \geq 3$ (see [4, Fig.1] and the full version of this paper for examples). Also, note that the $\frac{1}{2}\,\sigma(V)$ lower bound for k-ECSS does not apply for the version where multi-edges are allowed. In more detail, if multi-edges are allowed, then there exist k-edge connected graphs H such that any k-node connected graph has cost $\geq \Theta(k)\,c(H)$. See the full paper for more details.

THEOREM 3. *There is a polynomial-time algorithm for computing a solution to the metric-cost subset $[k,\ 1.5k]$-node-connectivity problem of cost*

$$\leq O(1)\cdot(\sigma(T) + \frac{k}{2}\ mst(T)) \leq O(1)\cdot \text{OPT}.$$

Remark: A loose analysis gives a constant factor between 800 and 1000 in the above theorem. Possibly, an approximation guarantee of ≤ 100 can be obtained by some changes to the algorithm. We have not attempted to optimise the constants in the approximation guarantees.

THEOREM 4. *There is a polynomial-time algorithm for computing a solution to the metric-cost NC-SNDP of cost $\leq O(\log r_{max})\cdot$* OPT.

Previous work

Over the past few decades, there has been significant research on approximation algorithms for network design. For early work in network design, see for example Dantzig, Ford & Fulkerson [11]. A celebrated and still unsurpassed result was Christofides' $\frac{3}{2}$-approximation algorithm for the metric-cost TSP [8]. Partly motivated by Christofides' result, there followed a stream of research on related problems in the design of metric-cost networks. Most of this research focused on small connectivity requirements, such as 2-edge connectivity and 2-node connectivity; see Frederickson & Ja'Ja' [13], Monma & Shallcross [25], Monma, Munson & Pulleyblank [24], and Bienstock, Brickell & Monma [4]. For constant k, this last paper gives a constant-factor approximation algorithm for k-NCSS. Moreover, the proof also shows that for metric graphs and any constant k, there exists a

k-node connected spanning subgraph of K_n whose cost is within a constant factor of the cost of any k-edge connected spanning subgraph, see [4, Sec.4]. They left open the question of extending these results to all k. This was followed by another burst of research, partly initiated by the work of Goemans & Bertsimas [14] who presented a logarithmic approximation algorithm for a general model called the edge-connectivity survivable network design problem (EC-SNDP) assuming metric costs. Soon after this, the research focus changed from metric costs to the more general setting of non-negative costs. Agrawal, Klein & Ravi [1], and Goemans & Williamson [16] built on the primal-dual method to obtain $O(1)$-approximation algorithms for some special cases of EC-SNDP with small (i.e., zero and one) connectivity requirements. Later, these methods were generalized to EC-SNDP, albeit with a logarithmic approximation guarantee, by Goemans et al. [15] based on work by Williamson et al. [30]. This line of research culminated with a 2-approximation algorithm for EC-SNDP by Jain [17].

Although there was considerable interest in extending these methods to the setting of node connectivity, there was limited success even for rather special cases of NC-SNDP. We mention a few results and refer the interested reader to [6] for more references. For the case of non-negative edge costs, Kortsarz & Nutov [21] and [7] have logarithmic (or worse) approximation guarantees for the k-NCSS problem. For metric costs, there is an $O(1)$-approximation algorithm due to Khuller & Raghavachari [19], and there are other related results in [5, 22]. Some explanation for this lack of good approximation algorithms for NC-SNDP comes from the recent hardness-of-approximation results of Kortsarz, Krauthgamer & Lee [20]. Also, see the surveys by Frank [12], Khuller [18], and Stoer [27], and the book by Vazirani [29].

We briefly mention the relationship between our work and the stream of exciting recent results on PTAS's (polynomial-time approximation schemes) for related problems. Beginning with the results of Arora [2] on the Euclidean TSP, many PTAS's have been obtained for problems in "geometric network design" where the edge costs come from special metrics such as the Euclidean metric, see [9, 10, 26] and the references in those papers. But, modulo P≠NP, such PTAS's do not exist in the setting of interest to us, namely, (general) metric costs; this follows from APX-hardness results in [3, 20, 28].

The rest of the paper is structured as follows. In Section 2, we discuss some preliminaries, and give an overview of our method for the metric-cost subset k-node connectivity problem. We present a constant-factor approximation algorithm for the problem in Section 3. Section 4 gives a constant-factor approximation algorithm for the metric-cost subset $[k, 1.5k]$-node connectivity problem. This leads to an $O(\log r_{\max})$-approximation algorithm for the metric-cost NC-SNDP in Section 5.

2. PRELIMINARIES AND AN OVERVIEW OF THE ALGORITHM FOR SUBSET K-CONNECTIVITY

Apart from Section 1, we omit the word 'node' from terms such as 'node-connectivity' when there is no danger of ambiguity.

Let the input graph be $G = (V, E)$. We denote the nodes by numbers $i = 1, 2, \ldots, n$, and for nodes i, j the edge between them is denoted ij. The cost of an edge $ij \in E$ is denoted c_{ij} or $c(i, j)$. The costs are said to be *metric* if the triangle inequality holds: $c(v, w) \leq c(v, u) + c(u, w)$, $\forall u, v, w \in V$. Whenever we assume metric costs, we also assume that G is the complete graph. Let k be an integer such that $n > k \geq 1$ (k may be a function of n). For a pair of nodes i, j, let $\kappa(i, j)$ denote the maximum number of openly node-disjoint i, j-paths. Recall that T denotes the set of terminal nodes. We use n' to denote $|T|$, and we assume $T = \{1, \ldots, n'\}$.

Let us formalize the lower bounds (i) and (ii) for the subset $[k,\ 1.5k]$-connectivity problem stated in Section 1. For each terminal node i, let Γ_i denote the set of r_i nearest neighbours of i; by convention, $i \notin \Gamma_i$. (Thus $|\Gamma_i| = r_i$ and $\forall x \in \Gamma_i, y \notin \Gamma_i \cup \{i\},\ c_{iy} \geq c_{ix}$.) Then note that σ_i denotes $\sum_{x \in \Gamma_i} c_{ix}$. Also, for each terminal node i, let μ_i denote σ_i / r_i, namely, the average cost of an edge from i to one of its r_i nearest neighbours. Note that each terminal node i has at least r_i neighbours in an optimal subgraph, thus $\text{OPT} \geq \frac{1}{2}\sigma(T)$. This gives the first lower bound. Next, we claim that $\text{OPT} \geq \frac{k}{2} mst(T)$. In more detail, we have $\text{OPT} \geq \frac{1}{2}\text{ECOPT}(T, 2k) \geq \frac{k}{2} mst(T)$, where $\text{ECOPT}(T, \lambda)$ denotes the minimum cost of a λ-edge connected subgraph of $G[T]$ (allowing multi-edges). To see this, start with a graph corresponding to OPT, and take two copies per edge to get an Eulerian multi-graph H' that is $2k$-edge connected on T, then apply the Lovász-Mader splitting-off theorem [23, Ex.6.51], [12], to eliminate all nodes of $V - T$ from H' to get a $2k$-edge connected multi-graph on the node set T that has cost $\geq \text{ECOPT}(T, 2k)$; then we apply the well-known fact that $\text{ECOPT}(T, \lambda) \geq \frac{\lambda}{2} mst(T)$. For metric costs, splitting off edges does not increase the cost. This gives the second lower bound: $\text{OPT} \geq \frac{k}{2} mst(T)$.

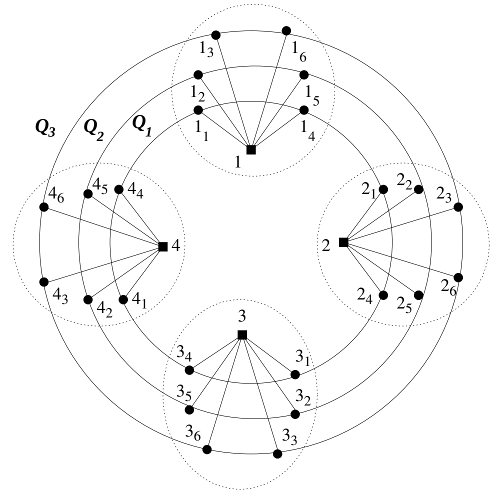

Figure 1: A key special case of the algorithm. Here, $k = 6$, $T = \{1, 2, 3, 4\}$, and the sets $\{i\} \cup \Gamma_i$ (indicated by dotted blobs) for $i \in T$ are pairwise disjoint. The tracks Q_1, Q_2, Q_3 are indicated by circles.

We first give an overview of our method for subset k-connectivity by describing a key special case where k is even, say $k = 2\ell$, and the sets $\{i\} \cup \Gamma_i$ of the terminals i are pairwise disjoint (that is, $(\{i\} \cup \Gamma_i) \cap (\{j\} \cup \Gamma_j) = \emptyset,\ \forall i \neq j \in T$). Arbitrarily name the nodes in Γ_i as

$i_1, i_2, \ldots, i_k$, $\forall i \in T$. Construct a cheap cycle Q on the terminals using the well-known MST-doubling heuristic for the TSP. (Start with an MST of the subgraph induced by T, replace each edge by two copies, and shortcut the resulting connected Eulerian graph to get a cycle Q with $V(Q) = T$ and $c(Q) \leq 2mst(T)$.) Let the sequence of terminals on Q be $1, 2, \ldots, n', 1$ (renumber the nodes if needed). For each $\tau = 1, \ldots, \ell$, construct a cycle $Q_\mathcal{T}$ "parallel" to Q where $Q_\mathcal{T} = 1_\mathcal{T}, 1_{\ell+\mathcal{T}}, 2_\mathcal{T}, 2_{\ell+\mathcal{T}}, 3_\mathcal{T}, 3_{\ell+\mathcal{T}}, \ldots, n'_\mathcal{T}, n'_{\ell+\mathcal{T}}, 1_\mathcal{T}$. (Informally, start with the cycle $1_\mathcal{T}, 2_\mathcal{T}, \ldots, n'_\mathcal{T}, 1_\mathcal{T}$, then for each $i = 1, \ldots, n'$ insert the node $i_{\ell+\mathcal{T}}$ between nodes $i_\mathcal{T}$ and $(i+1)_\mathcal{T}$; see Figure 1;) Let us refer to these cycles as *tracks*. It can be seen that a track $Q_\mathcal{T}$ has cost $c(Q_\mathcal{T}) \leq c(Q) + \sum_{i=1}^{t} 2(c(i, i_\mathcal{T}) + c(i, i_{\ell+\mathcal{T}}))$ (see the second subroutine below), and the total cost of the tracks is $\sum_{\mathcal{T}=1}^{\ell} c(Q_\mathcal{T}) \leq \ell \cdot c(Q) + 2\sigma(T)$. Finally, for each terminal $i \in T$, we add the k edges $ii_1, ii_2, \ldots, ii_k$. The resulting subgraph is our solution graph H; it has cost $c(H) \leq 2\ell \cdot mst(T) + 3\sigma(T) \leq 2\text{OPT} + 6\text{OPT} = 8\text{OPT}$. Note that each terminal has precisely two neighbours in each track. Thus H satisfies the connectivity requirements, because for every pair of terminals $i, j (i \neq j)$, each of the $k/2$ tracks contributes 2 openly disjoint i, j paths.

The algorithm uses the following two subroutines. Note that the solution graph H is simple, so when we add edges to H we do so without creating multi-edges.

- The first subroutine *copies a specified set of neighbours* of a terminal i to another terminal v (possibly, v is adjacent to i). More precisely, given a terminal i and a specified set of neighbours of i, call it N_i, and another terminal v, the subroutine adds an edge vx to H for each node $x \in N_i$ (without creating multi-edges or loops in H). After this step, $\kappa(i, v) \geq |N_i|$ in H. The cost of the new edges is $\leq |N_i|\, c(i, v) + \sum_{x \in N_i} c(i, x)$; moreover, if there is a positive real number γ such that $\sum_{x \in N_i} c(i, x) \leq \gamma \sigma_i$, then the cost of the new edges is $\leq |N_i|\, c(i, v) + \gamma \sigma_i$.
- The other subroutine starts with a cycle containing a terminal i and *inserts new node(s) into the cycle*. Given a cycle Q', a terminal i in Q', and a node $x \notin V(Q')$, we first add two copies of the edge ix to Q' to get a connected Eulerian graph. Then we shortcut this Eulerian graph (as in the MST-doubling heuristic for the TSP) to obtain a new cycle Q with node set $V(Q') \cup \{x\}$. The increase in cost is $\leq 2c(i, x)$.

It is important for our analysis to get good upper bounds on the costs of the tracks. Note that the tracks are pairwise node disjoint; thus each terminal is in at most one track. But, for upper-bounding the track costs, we use the following accounting trick. Consider any track $Q_\mathcal{T}$. We assume that the track initially consists of all the terminals, thus $V(Q_\mathcal{T}) = T$, and using the MST-doubling heuristic we have $c(Q_\mathcal{T}) \leq 2mst(T)$. Subsequently, the algorithm may insert new nodes into the track – such insertions occur while we are processing some terminal – thus for inserting node x while processing terminal i the cost $c(Q_\mathcal{T})$ increases by $\leq 2c(i, x)$. Possibly, x may be another terminal – in that case, we implicitly remove x from $Q_\mathcal{T}$ and then insert x via the double-edge ix. At the end of the execution, we keep only those terminals that were explicitly inserted into $Q_\mathcal{T}$ and remove all the other terminals from $Q_\mathcal{T}$; clearly, this does not increase the cost $c(Q_\mathcal{T})$. Note that this "historical view" of $Q_\mathcal{T}$ is only needed for upper-bounding the cost. Other than this, it may be easier to view the tracks as being pairwise node disjoint all through the execution, and this is the viewpoint we use in presenting the detailed algorithm.

3. THE ALGORITHM FOR SUBSET K-CONNECTIVITY

This section is devoted to an algorithm and proof for Theorem 1. The detailed algorithm follows. An analysis of the cost of the edges added to H (the solution graph) is given after the algorithm. A terminal may be in two states *active* or *inactive*. Initially, all the terminals are active. Let ℓ denote $\lceil k/2 \rceil$. Initially, H is the graph consisting of all the terminal nodes and no edges, thus $H = (T, \emptyset)$.

(1) [DE-ACTIVATE TERMINALS AND CONSTRUCT DISJOINT BALLS FOR ACTIVE TERMINALS]
Renumber the terminals as $1, 2, \ldots, n'$ by increasing value of μ; thus $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_{n'}$.
Note: $\mu_h \leq \mu_j$ iff $\sigma_h \leq \sigma_j$.
Scan the terminals in the order $1, 2, \ldots, n'$, and skip the current terminal if it is inactive. For an active terminal i, construct the set $B_i = \{j \mid c(i, j) \leq \alpha\mu_i\}$, where we choose $\alpha = 2$. For each active terminal $v > i$, if $c_{iv} \leq (\alpha\mu_i + \beta\alpha\mu_v)$, where we fix $\beta = 2$, then make v inactive, and record i as the parent of v by assigning $p(v) = i$. (The aim is to ensure that the sets B_i of active terminals i are pairwise disjoint.)

Note that $i \in B_i$ and $|B_i| \geq 1 + (1 - \frac{1}{\alpha})k = 1 + \frac{k}{2}$. (Otherwise, we have $\geq k/\alpha = k/2$ nodes x in Γ_i with $c(i, x) > \alpha\mu_i = \alpha\sigma_i/k$, so these nodes contribute $> \sigma_i$ to $\sum_{x \in \Gamma_i} c(i, x)$.) Hence, $|B_i - \{i\}| \geq \ell$. Also note that $\mu_{p(v)} \leq \mu_v$ for each inactive terminal v.

Choose the ℓ nodes in B_i nearest to i and name them as $i_1, i_2, \ldots, i_\ell$ such that $c(i, i_1) \leq c(i, i_2) \leq \cdots \leq c(i, i_\ell)$.

(2) [CONSTRUCT ℓ TRACKS ON THE DISJOINT BALLS]
After step (1), let T^* denote the set of active terminals and let $n^* = |T^*|$. If $n^* < 3$, then apply step (2′) and stop. Otherwise, construct a cheap cycle Q on the active terminals by applying the MST-doubling heuristic for the TSP to the subgraph induced by T^*. Renumber the terminals such that $Q = 1, 2, \ldots, n^*, 1$, that is, the active terminals get the numbers in $\{1, \ldots, n^*\}$ according to their ordering in Q. Construct ℓ tracks $Q_1, Q_2, \ldots, Q_\ell$, where track $Q_\mathcal{T} = 1_\mathcal{T}, 2_\mathcal{T}, \ldots, n^*_\mathcal{T}, 1_\mathcal{T}$ $(\tau = 1, \ldots, \ell)$. Add all the tracks (but not the cycle Q) to H. The cost of the tracks constructed in this step is analysed in Proposition 6 below.

(2′) [SPECIAL HANDLING FOR 1 OR 2 ACTIVE TERMINALS]
Skip this step if $n^* \geq 3$. Suppose $n^* = 1$. Let the active terminal be i. Add all the edges $iv, v \in \Gamma_i$, and then for each inactive terminal j, copy the set Γ_i of neighbours of i to j. The resulting graph H satisfies the connectivity requirements.

Suppose $n^* = 2$. Let the active terminals be h, i, with $\sigma_h \leq \sigma_i$. Add all the edges $hq, q \in \Gamma_h$, and $iv, v \in \Gamma_i$. Then add a matching M of maximum size between the nodes in $\Gamma_i - (\Gamma_h \cup \{h\})$ and in $\Gamma_h - (\Gamma_i \cup \{i\})$; now, each matching edge qv (say $q \in \Gamma_h - \{i\}$ and $v \in \Gamma_i - \{h\}$)

gives an h,i path, namely, h,q,v,i. Finally, for each inactive terminal j, copy the set $\Gamma_{p(j)}$ of neighbours of $p(j)$ to j. The resulting graph H satisfies the connectivity requirements.

(3) [AUGMENT DISJOINT BALLS AND ASSIGN TOKEN ARCS] In summary, this step scans the active terminals i and augments each "ball" B_i to get an "augmented ball" B'_i (that ideally has $|B'_i| \geq r_i + 1 = k+1$) such that these augmented balls are pairwise disjoint. The obvious construction for B'_i is to start with B_i and then add some nodes from $\Gamma_i - B_i$, but then the augmented balls may intersect. We "de-intersect" two intersecting sets B'_h and B'_i, while preserving the balls B_h and B_i, by assigning so-called *token arcs* to the active terminals such that for each active terminal i, $|B'_i|$ plus the number of token arcs assigned to i is $\geq r_i + 1 = k+1$. Consider one special case: suppose that h and i are active terminals and node q is in $B'_h \cap B'_i$ but $q \notin B_h \cup B_i$. Then we compare the costs of the edges hq and iq and "replace" the costlier edge, say iq, by a token arc whose cost we fix to be $3c_{iq}$; that is, we remove q from B'_i and instead assign to i a token arc with cost $3c_{iq}$. The details follow.

Renumber the terminals so that the active terminals i in order of increasing μ_i values are $1, 2, \ldots, n^*$, and scan the active terminals in this order. Start the scan of $i \in T^*$ by defining $B'_i := B_i$ if $\Gamma_i \subseteq B_i$, and $B'_i := \Gamma_i$ otherwise. If B'_i is disjoint from B'_h for all active terminals $h < i$, then continue with the next active terminal, otherwise, for each active terminal $h < i$ with $B'_h \cap B'_i \neq \emptyset$, examine the nodes q in $B'_h \cap B'_i$ in any order. Note that $\mu_h \leq \mu_i$.

(a) Suppose $q \in B_h$. Then note that $q \notin B_i$ and $c_{iq} > \alpha\mu_i \geq \alpha\mu_h$. Remove q from B'_i and give to i a token arc (i,h) with cost $3c_{iq}$. (Later, this token arc will be replaced by an edge ix where $x \in B_h$; note that the cost of ix is $\leq c_{iq} + c_{hq} + c_{hx} \leq c_{iq} + 2\alpha\mu_h \leq 3c_{iq}$.)

(b) Otherwise, $q \in B'_h - B_h$. Suppose $q \in B_i$. Then note that $c_{hq} + c_{iq} \geq c_{hi} \geq (\alpha\mu_h + \beta\alpha\mu_i)$ (the last inequality holds because both h,i are active), and $c_{iq} \leq \alpha\mu_i$, hence, $c_{hq} \geq \alpha(\mu_h + \mu_i)$ (recall that $\beta = 2$). Remove q from B'_h and give to h a token arc (h,i) with cost $3c_{hq}$. (Later, this token arc will be replaced by an edge hx, $x \in B_i$, of cost $\leq c_{hq} + c_{iq} + c_{ix} \leq c_{hq} + 2\alpha\mu_i \leq c_{hq} + 2c_{hq} \leq 3c_{hq}$.)

(c) Suppose $q \in B'_h - B_h$ and $q \in B'_i - B_i$. Then we compare c_{iq} and c_{hq}.

If $c_{iq} \geq c_{hq}$, then remove q from B'_i and give to i a token arc (i,h) with cost $3c_{iq}$. (Later, this token arc will be replaced by an edge ix, $x \in B_h$, of cost $\leq c_{iq} + c_{hq} + c_{hx} \leq 2c_{iq} + \alpha\mu_h \leq 3c_{iq}$, where the last inequality holds because $c_{iq} > \alpha\mu_i \geq \alpha\mu_h$.)

Otherwise, we have $c_{iq} < c_{hq}$. Then we remove q from B'_h and give to h a token arc (h,i) with cost $3c_{hq}$. (Later, this token arc will be replaced by an edge hx, $x \in B_i$, of cost $\leq c_{hq} + c_{iq} + c_{ix} \leq 2c_{hq} + \alpha\mu_i \leq 3c_{hq}$, where the last inequality holds because $c_{hq} + c_{iq} \geq c_{hi} \geq (\alpha\mu_h + \beta\alpha\mu_i)$ (as in (b) above), hence, $c_{hq} \geq \frac{\alpha}{2}(\mu_h + \beta\mu_i) \geq \alpha\mu_i$ (recall that $\beta = 2$).)

After step (3), note that the cost of a token arc (i,j) depends on the cost of the associated edge iq and is $3c_{iq}$.

(4) [ATTACH ACTIVE TERMINALS TO TRACKS] In summary, this step scans each active terminal i and adds edges from i to the tracks such that each track $Q_\tau, \tau = 1, \ldots, \lfloor k/2 \rfloor$, gets two neighbours of i, and the last track Q_ℓ gets ≥ 1 neighbour of i.

First add edges from i to each of $i_1, i_2, \ldots, i_\ell$; also, mark the nodes $i_1, i_2, \ldots, i_\ell$ as used.

Then for each $\tau = 1, 2, \ldots, \lfloor k/2 \rfloor$, do the following. If an unused token arc (i,h) is available, then choose it, mark it as used, and add the edge ih_τ; note that h_τ is in B_h and is the "first neighbour" of h in track Q_τ; also, note that $c(i, h_\tau)$ is $\leq$ the cost of the token arc (i,h). If no unused token arcs are available, then choose an unused node $q \in B'_i$, mark it as used, insert q into track Q_τ, and add the edge iq. (Note that the number of token arcs given to i plus $|B'_i|$ is $\geq k+1$, hence, this step will find $\lfloor k/2 \rfloor$ token arcs or unused nodes, excluding the nodes $i_1, i_2, \ldots, i_\ell$.)

For each active terminal i, let N_i denote the set of neighbours of i in the tracks, just after step (4) is applied to i.

(5) [ATTACH INACTIVE TERMINALS TO TRACKS] Finally, "attach" the inactive terminals to the tracks. Note that an inactive terminal may be already in one of the tracks. For each inactive terminal j, copy the set of neighbours $N_{p(j)}$ of the parent $p(j)$ to j.

PROPOSITION 5. *The graph constructed in step (2′) has cost* ≤ 16OPT.

PROOF. Suppose $n^* = 1$, and let i be the (unique) active terminal. Then $c(H) \leq \sigma_i + \sum_{j \in T - T^*}(kc_{ij} + \sigma_i) \leq \sigma_i + \sum_{j \in T-T^*}(k(\alpha\mu_i + \beta\alpha\mu_j) + \sigma_i) \leq \sigma_i + \sum_{j \in T-T^*}(\alpha(1+\beta)\sigma_j + \sigma_j) \leq 7\sigma(T) \leq 14$OPT (we have $\alpha = 2, \beta = 2$, and we used $\sigma_i \leq \sigma_j$ for an inactive terminal j).

Suppose $n^* = 2$, and let i,h be the two active terminals. Then recall that M denotes a matching of maximum size between the nodes in $\Gamma_i - (\Gamma_h \cup \{h\})$ and in $\Gamma_h - (\Gamma_i \cup \{i\})$; note that an edge $qv \in M$ (say, $q \in \Gamma_h, v \in \Gamma_i$) has cost $\leq c_{hq} + c_{hi} + c_{iv}$, hence, $c(M) \leq \sigma_h + \sigma_i + k \cdot mst(T)$; the other edges in H contribute a cost of $\leq \sigma_h + \sigma_i + \sum_{j \in T-T^*}(\alpha(1+\beta)\sigma_j + \sigma_j)$ (as in the analysis for $n^* = 1$) hence, $c(H) \leq 7\sigma(T) + k \cdot mst(T) \leq 16$OPT. □

PROPOSITION 6. *(i) The total cost of the edges added by step (4) and incident to an active terminal i is* $\leq 4\sigma_i$. *(ii) At the end of step (4), the total cost of the* ℓ *tracks is*

$$\leq 2\ell \cdot mst(T) + 4\sigma(T^*).$$

PROOF. For an active terminal i, the total cost of the token arcs (i,h) given to i is $\leq 3\sigma_i$. The cost of the edges added that are incident to i, but excluding the cost due to the token arcs, is $\leq \alpha\sigma_i$ if $|B_i| \geq k+1$ (in this case, $B'_i = B_i$ and no token arcs are given to i), and is $\leq \sum_{x \in \Gamma_i} c_{ix} \leq \sigma_i$ otherwise. Thus the total cost of the added edges incident to i is $\leq \max(\alpha\sigma_i,\ \sigma_i + 3\sigma_i) \leq 4\sigma_i$.

The total cost of the ℓ tracks (that were constructed in step (2) and modified in step (4)) is $\leq 2\ell \cdot mst(T) + 4\sigma(T^*)$.

To see this, first consider the term $2\ell \cdot mst(T)$. Recall (from Section 2) the accounting trick we use for upper-bounding the cost of a track; due to this, we take the upper bound on the cost of Q (the cheap cycle on T^* in step (2)) to be $2mst(T)$ rather than $2mst(T^*)$. Summed over ℓ tracks, this gives $2\ell \cdot mst(T)$. For the second term, note that $i \in T^*$ contributes $\le \sum_{q \in B'_i} 2c(i,q)$, and this is $\le 2k(\alpha\mu_i)$ if $\Gamma_i \subseteq B_i$ (then $B'_i = B_i$), and $\le 2\sigma_i$ otherwise (then $B'_i = \Gamma_i$). □

PROPOSITION 7. *The edges added by step (5) and incident to the inactive terminals have total cost* $\le 10\sigma(T - T^*)$.

PROOF. Suppose the cost of the added edges incident to an active terminal i is $\le \gamma\sigma_i$. (From Proposition 6, we have $\gamma = 4$.) Then the cost of the edges added for an inactive terminal j with parent i is $\le k \cdot c_{ij} + \gamma\sigma_i \le k(\alpha\mu_i + \beta\alpha\mu_j) + \gamma\sigma_i \le (\alpha(\beta+1)+\gamma)\sigma_j$, using the fact that $\sigma_{p(j)} \le \sigma_j$. Thus the total cost of the edges added in this step is $\le 10\sigma(T - T^*)$, using $\alpha = 2, \beta = 2, \gamma = 4$. □

PROOF OF THEOREM 1. By the above propositions, the total cost of H is $\le 2\ell \cdot mst(T) + 4\sigma(T^*) + \gamma\sigma(T^*) + 10\sigma(T - T^*) \le (k+1)mst(T) + 8\sigma(T^*) + 10\sigma(T - T^*) \le (k+1)mst(T) + 10\sigma(T) \le (2 + \frac{2}{k})\text{OPT} + 20\text{OPT} \le 24\text{OPT}$.

We claim that the graph H has the required connectivity property, namely, $\kappa(i,j) \ge k, \forall i \ne j \in T$. To see this, consider any pair of terminals i,j and consider any one track Q_τ. Suppose that either i is in Q_τ, or i is not in Q_τ but has two neighbours in Q_τ. Suppose the same statement holds for j (that is, j is in Q_τ, or j is not in Q_τ but has two neighbours in Q_τ). Then, Q_τ (together with the edges from i and j to Q_τ) contributes two openly disjoint i,j paths. Similarly, Q_τ contributes one i,j path if both i and j either are in Q_τ or have a neighbour in Q_τ. By construction, each active terminal has two neighbours in each of the tracks Q_τ for $\tau = 1, \ldots, \lfloor k/2 \rfloor$, and has a neighbour in Q_ℓ; similarly, each inactive terminal is either in Q_τ or has two neighbours in Q_τ for $\tau = 1, \ldots, \lfloor k/2 \rfloor$, and is in Q_ℓ or has a neighbour in Q_ℓ. Then, for any two terminals i and j, H has k openly disjoint i,j paths, since each of the tracks Q_τ for $\tau = 1, \ldots, \lfloor k/2 \rfloor$, contributes two openly disjoint i,j paths, Q_ℓ contributes an i,j path, and these k paths together are openly disjoint. □

4. THE ALGORITHM FOR SUBSET [K, 1.5K]-CONNECTIVITY

In this section, we extend the methods of the previous section to obtain an $O(1)$-approximation algorithm for the the subset $[k, 1.5k]$-connectivity problem. The main contribution of this section is the following result concerning a restricted case of Theorem 3. Due to space constraints, we defer the proof to the full paper.

THEOREM 8. *Let k be an integer multiple of 4, thus $k = 0$* (mod 4). *There is polynomial-time algorithm for computing a solution to the metric-cost subset $[k,\ 1.5k]$-connectivity problem of cost $\le O(1) \cdot$ OPT.* □

Remark: A loose analysis gives a constant factor between 800 and 900 in the above theorem.

Theorem 3 follows by combining this theorem with Theorem 1. To see this, suppose that $k \ne 0$ (mod 4) (otherwise, we are done). Let $\hat{k} \ge k$ denote the next integer multiple of 4; clearly, $\hat{k} - k \le 3$. Then for each $\rho = k, k+1, \ldots, \hat{k}-1$, we apply the algorithm in Theorem 1 to the following instance $\Pi(\rho)$ of the subset ρ-connectivity problem to obtain a solution subgraph $H(\rho)$: we take the requirement of a terminal i in $\Pi(\rho)$ to be $r'_i = 0$ if $r_i < \rho$, and we take $r'_i = \rho$ if $r_i \ge \rho$; the rest of the instance stays the same. Finally, we apply the algorithm of this section to the instance of subset $[\hat{k},\ 1.5\hat{k}]$-connectivity where we take the requirement of a terminal i to be $r'_i = 0$ if $r_i < \hat{k}$, and we take $r'_i = r_i$ if $r_i \ge \hat{k}$; the rest of the instance stays the same. Let H' be solution subgraph. Then, for the original instance (of subset $[k,\ 1.5k]$-connectivity), we output the solution subgraph $H^* = H(k) \cup H(k+1) \cup \cdots \cup H(\hat{k}-1) \cup H'$ whose cost is at most $O(1)$OPT. To see that H^* satisfies the connectivity requirements, note that for every pair of terminals i,j, one of the subgraphs forming H^* (namely, one of $H(k), H(k+1), \ldots, H(\hat{k}-1), H'$) has $\min(r_i, r_j)$ openly disjoint i,j-paths.

5. THE ALGORITHM FOR NODE-CONNECTIVITY SNDP

This section presents a proof of Theorem 4, based on (the algorithms in) Theorems 1, 3. For the sake of motivation, let us obtain an $O(\log r_{max})$-approximation algorithm for a restricted version of NC-SNDP where every terminal has a requirement r_i and every pair of terminals i,j has the requirement $r_{i,j} = \min(r_i, r_j)$. The method is similar to the method for proving Theorem 3 from Theorems 1 and 8.

Let OPT denote the optimal value of the instance (of restricted NC-SNDP). First, for each $\rho = 1, 2, \ldots, 7$, we apply the algorithm in Theorem 1 to the following instance $\Pi(\rho)$ of the subset ρ-connectivity problem to obtain a solution subgraph $H(\rho)$: we take the requirement of a terminal i in $\Pi(\rho)$ to be $r'_i = 0$ if $r_i < \rho$, and we take $r'_i = \rho$ if $r_i \ge \rho$; the rest of the instance stays the same. By Theorem 1, the cost of $H(\rho)$ is $O(1) \cdot$ OPT. After this, we repeatedly apply the algorithm in Theorem 8 to solve an instance (specified below) of subset $[\rho,\ 1.5\rho]$-connectivity, where ρ is an integer multiple of 4 ($\rho = 8, 12, 16, 24, \ldots$, details later), to obtain a solution subgraph $H'(\rho)$. The instances of subset $[\rho,\ 1.5\rho]$-connectivity are as follows: we take the requirement of a terminal i to be $r'_i = 0$ if $r_i < \rho$, we take $r'_i = r_i$ if $\rho \le r_i \le 1.5\rho$, and we take $r'_i = 1.5\rho$ if $r_i > 1.5\rho$. By Theorem 8, the cost of $H'(\rho)$ is $O(1) \cdot$ OPT. We start with $\rho = 8$, and we iterate until $r_{max} \le 1.5\rho$; after each iteration, we update ρ to the largest integer multiple of 4 that is ≤ 1.5 times the previous ρ. Clearly, the number of iterations is $O(\log r_{max})$. Finally, we output the solution subgraph H^* for the instance (of restricted NC-SNDP); H^* is the union of all the solution subgraphs $H(\rho)$, $\rho = 1, \ldots, 7$, and $H'(\rho)$, $\rho = 8, 12, \ldots$. Thus H^* is the union of $O(\log r_{max})$ subgraphs such that each of these subgraphs has cost $O(1) \cdot$ OPT, and so H^* has cost $O(\log r_{max}) \cdot$ OPT. To see that H^* satisfies the connectivity requirements, note that for every pair of terminals i,j, one of the subgraphs forming H^* has $\min(r_i, r_j)$ openly disjoint i,j-paths, namely, the subgraph $H(\min(r_i, r_j))$ if $\min(r_i, r_j) \le 7$, otherwise, any subgraph $H'(\rho)$ where ρ satisfies $\rho \le \min(r_i, r_j) \le 1.5\rho$.

Our algorithm for metric-cost NC-SNDP is similar to the algorithm described above for the restricted version of NC-SNDP. Let Π^* be an instance of NC-SNDP, and let OPT denote its optimal value. We use k^f to denote an integer multiple of 4 such that $r_{max} \leq 1.5k^f$. We repeatedly apply the algorithm of Theorem 1 (for subset k-connectivity) for $k = 1, \ldots, 7$, and derived instances $\Pi(1), \ldots, \Pi(7)$ to obtain solution subgraphs $H(1), \ldots, H(7)$. Then we repeatedly apply the algorithm of Theorem 8 (for subset $[k,\ 1.5k]$-connectivity) for $k = 8, 12, 16, 24, \ldots, k^f$ and derived instances $\Pi'(8), \Pi'(12), \ldots, \Pi'(k^f)$ to obtain solution subgraphs $H'(8),\ H'(12), \ldots, H'(k^f)$. We start these iterations with $k = 8$, and we iterate until $k = k^f$; after each iteration, we update k to the largest integer multiple of 4 that is ≤ 1.5 times the previous k. The construction of the derived instances $\Pi(\rho)$ and $\Pi'(k)$ is described below.

Finally, we output the solution subgraph H^* for Π^*; H^* is the union of all the solution subgraphs $H(k)$, $k = 1, \ldots, 7$, and $H'(k)$, $k = 8, 12, \ldots, k^f$; we call these solution subgraphs the *constituent subgraphs* of H^*. Below, we prove that the cost of each of the constituent subgraphs is at most $O(1) \cdot \text{OPT}$. Clearly, the number of iterations is $O(\log r_{max})$. Thus H^* is the union of $O(\log r_{max})$ subgraphs such that each of these subgraphs has cost $O(1) \cdot \text{OPT}$, and so H^* has cost $O(\log r_{max}) \cdot \text{OPT}$. Below, we prove that H^* satisfies the connectivity requirements, because for every pair of terminals i, j, one of the constituent subgraphs of H^* has $\geq r_{i,j}$ openly disjoint i, j-paths.

We define the derived instances via a well-studied problem in network design, namely, the *generalized Steiner tree* problem, which is as follows: we are given a graph $G = (V, E)$, edge costs c, and $\hat{q}$ sets of terminal nodes $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{\hat{q}}$; the goal is to compute an (approximately) minimum-cost forest F of G such that each terminal set $\hat{D}_m, m = 1, \ldots, \hat{q}$, is contained in a (connected) component of F. Goemans and Williamson [16], based on earlier work by Agrawal et al. [1], gave 2-approximation algorithms for this problem based on the primal-dual method.

Here is the construction for one of the derived instances $\Pi'(k)$; recall that this is an instance of the subset $[k, 1.5k]$-connectivity problem, where k is a fixed parameter. We start from Π^* and construct a requirements graph R with node set T and edge set $E(R)$ as follows. For each terminal pair i, j with $k \leq r_{i,j} \leq 1.5k$ (i.e., the requirement for the pair is within the valid range for our derived instance), we add the edge ij to R. Denote the node sets of the (connected) components of R by $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{\hat{q}}$. Next, we define an instance $\Pi(gst)$ of the *generalized Steiner tree* problem on the graph G with edge costs c (here, G, c are as in Π^*), and with terminal sets $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{\hat{q}}$. We solve this auxiliary problem $\Pi(gst)$ by applying the primal-dual algorithm of Goemans and Williamson [16]. Let $F \subseteq E(G)$ be the forest computed by the Goemans-Williamson algorithm, and let $F_1, F_2, \ldots, F_q$ denote the partition of F into connected components. Let the set of terminals in the component of F_m be denoted by D_m, $m = 1, \ldots, q$; thus each set D_m is the union of one or more of the terminal sets $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{\hat{q}}$. For each $m = 1, \ldots, q$, we define an instance $\Pi'_m(k)$ of the subset $[k,\ 1.5k]$-connectivity problem as follows: the graph G and the edge costs c are as in Π^*; the set of terminal nodes is D_m, and the requirement r'_i of a terminal $i \in D_m$ is defined to be $\max(r_{i,j}\ :\ \{i, j\} \in E(R))$; clearly, $k \leq r'_i \leq 1.5k,\ \forall i \in D_m$. We take the derived instance $\Pi'(k)$ to be the disjoint union of these instances $\Pi'_m(k)$, $m = 1, \ldots, q$, i.e., we assume that each instance $\Pi'_m(k)$ has its own copy of G and c. To solve $\Pi'(k)$, we take each $m = 1, \ldots, q$, and apply the algorithm in Theorem 8 separately to $\Pi'_m(k)$ to obtain a solution subgraph, call it $H'_m(k)$. (These instances $\Pi'_m(k)$ are pairwise disjoint, and we solve them separately, one by one.) Then we take the union of the subgraphs $H'_1(k), \ldots, H'_m(k)$ and call it $H'(k)$; this is the solution subgraph of $\Pi'(k)$. The cost of the subgraphs $H'_m(k)$, $m = 1, \ldots, q$, is analysed below.

Our reasons for using the auxiliary problem $\Pi(gst)$ for defining the instance $\Pi'(k)$ may be seen from the following example. Suppose that k is large (say $k = \sqrt{n}$) and the edges in $E(R)$ form a matching say $\{\{s_1, t_1\}, \{s_2, t_2\}, \ldots, \{s_{\hat{q}}, t_{\hat{q}}\}\}$, say $\hat{q} = \Theta(n)$. Moreover, suppose that G has a cut $\delta(S)$ such that each edge in this cut is expensive, and some of the edges in $E(R)$ have both ends in S and the remaining edges in $E(R)$ have both ends in $V - S$. Say the optimal solution consists of two disjoint subgraphs, one contained in the subgraph induced by S and the other contained in the subgraph induced by $V - S$. Then we cannot take $\Pi'(k)$ to be a single instance with terminal set $\{s_1, \ldots, s_{\hat{q}}, t_1, \ldots, t_{\hat{q}}\}$, because then every solution subgraph will have $\geq k$ edges from the expensive cut $\delta(S)$. Also, we cannot take $\Pi'(k)$ to consist of $\hat{q}$ separate sub-instances with one sub-instance for each connected component of $R = (T, E(R))$, because the optimal values of these sub-instances may sum to $\hat{q} \cdot \text{OPT}$, and the solution subgraph computed by our algorithm may have cost as high as this (assuming that the algorithm returns the union of the solution subgraphs of these $\hat{q}$ sub-instances). We get around this difficulty by using the Goemans-Williamson algorithm to merge the connected components of $R = (T, E(R))$ into appropriate "clusters" and then we construct a separate sub-instance for each of these "clusters" (these are the sub-instances that we called $\Pi'_1(k), \ldots, \Pi'_q(k)$). The key point is that (i) these sub-instances have pairwise disjoint terminal sets $D_1, \ldots, D_q$, hence, the sum of the $\sigma()$ lower-bounds (used in Theorem 8), namely, $\sum_{m=1}^{q} \sigma(D_m)$, is $\leq$ the $\sigma()$ lower-bound of Π^*, and (ii) the following proof (which is based on the 2 approximation guarantee of Goemans and Williamson) shows that the sum of the *mst*() lower-bounds for these sub-instances, namely, $\sum_{m=1}^{q} mst(D_m)$, is $\leq O(1)$ times the *mst*() lower-bound of Π^*. Also, for each sub-instance, the solution subgraph has cost within an $O(1)$ factor of the sum of its $\sigma()$ and *mst*() lower-bounds. Hence, the union of the solution subgraphs of these sub-instances has cost within an $O(1)$ factor of the optimal value of Π^*.

The construction of the instances $\Pi(\rho)$, $\rho = 1, \ldots, 7$, is similar to that of the instances $\Pi'(k)$. We start with $R = (T, E(R))$ where $E(R)$ consists of terminal pairs $\{i, j\}$ with $r_{i,j} = \rho$. Then we obtain a family of pairwise disjoint sub-instances $\Pi_1(\rho), \Pi_2(\rho), \ldots$ and these sub-instances together form $\Pi(\rho)$.

PROOF OF THEOREM 4. Recall that Π^* denotes the instance of NC-SNDP, OPT denotes the optimal value of Π^*, and H^* denotes the solution subgraph of Π^* found by our algorithm. The goal is to analyze the cost of the constituent subgraphs of H^* and show that each has cost $\leq O(1) \cdot \text{OPT}$, and then to show that H^* satisfies the connectivity requirements. The proof is based on the following LP (linear programming) relaxation P^* of Π^* that interprets each requirement $r_{i,j}$ as a requirement for $r_{i,j}$ edge-disjoint i, j paths. Thus the optimal value of P^* gives a lower bound on OPT. The LP has a variable x_e, $0 \leq x_e \leq 1$, for each edge $e \in E$; the intention is that each feasible solution H of Π^* gives a zero-one vector

$x \in \mathbb{R}^E$ that satisfies two conditions: $x_e = 1$ iff $e \in H$, and x satisfies the constraints of the LP (though feasible zero-one solutions of the LP may not give feasible solutions of Π^*).

P^* :

$$\begin{array}{rcl} z^* & = & \min \sum_{e \in E} c_e x_e \\ \text{subject to} & & \\ x(\delta(S)) & \geq & \max\{r_{i,j} \ : \ i \in S, j \notin S\}, \quad \forall S \subseteq V \\ x_e & \geq & 0, \quad \forall e \in E \end{array}$$

Focus on one of the derived instances $\Pi'(k)$ and its associated generalized Steiner tree instance $\Pi(gst)$. We use the notation from the construction of $\Pi'(k)$ given above. Goemans and Williamson [16] proved that the cost of the forest computed by their algorithm is ≤ 2 times the optimal value $z(gst)$ of the following LP relaxation $P(gst)$ of $\Pi(gst)$. The LP has a variable x_e, $0 \leq x_e \leq 1$, for each edge $e \in E$; the intention is that each feasible solution F of $\Pi(gst)$ corresponds to a zero-one vector $x \in \mathbb{R}^E$ that satisfies two conditions: $x_e = 1$ iff $e \in F$, and x satisfies the constraints of the LP.

$P(gst)$:

$$\begin{array}{rcl} z(gst) & = & \min \sum_{e \in E} c_e x_e \\ \text{subject to} & & \\ x(\delta(S)) & \geq & 1, \quad \forall S \subseteq V : \exists m = 1, \ldots, \hat{q} : \\ & & \qquad \emptyset \neq S \cap \hat{D}_m \neq \hat{D}_m \\ x_e & \geq & 0, \quad \forall e \in E \end{array}$$

A key observation is that $k \cdot z(gst) \leq$ OPT. To see this, note that multiplying the right-hand-side of any constraint of the LP $P(gst)$ by k gives a constraint that is valid for the LP P^*. (This follows because whenever we have a constraint $x(\delta(S)) \geq 1$ in the LP $P(gst)$, then the node set S separates two terminals v, w such that the requirements graph R has an v, w-path consisting of terminal-pairs $\{i, j\}$ such that $r_{i,j} \geq k$; since the v, w-path of R "crosses" S, one of the terminal-pairs $\{i, j\}$ in the v, w-path "crosses" S, therefore, $\max\{r_{i,j} \ : \ i \in S, j \notin S\} \geq k$, hence, the constraint "$x(\delta(S)) \geq k$" is a valid constraint for the LP P^*.) Consequently, for every feasible solution x^* of the LP P^*, we see that $\frac{1}{k}x^*$ is a feasible solution of the LP $P(gst)$. Moreover, if x^* is an optimal solution of the LP P^*, then we have $z(gst) \leq \frac{1}{k}c(x^*) = \frac{1}{k}z^* \leq \frac{1}{k}$OPT, or equivalently, $k \cdot z(gst) \leq$ OPT.

Focus on the cost of the solution subgraph $H'(k) = H'_1(k) \cup H'_2(k) \cup \cdots \cup H'_q(k)$, and note that for each $m = 1, \ldots, q$ the cost of $H'_m(k)$ is $O(k) \cdot mst(D_m) + O(1) \cdot \sigma(D_m)$ (by Theorem 8), where D_m denotes the terminal set of $H'_m(k)$. Then the cost of $H'(k)$ is

$$\begin{array}{rl} & O(k) \cdot \displaystyle\sum_{m=1}^{q} mst(D_m) + O(1) \cdot \sum_{m=1}^{q} \sigma(D_m) \\ \leq & O(k) \cdot \displaystyle\sum_{m=1}^{q} c(F_m) + O(1) \cdot \sigma(T) \\ & \quad \text{(since } mst(D_m) \leq 2c(F_m), \ \forall m = 1, \ldots, q) \\ \leq & O(k) \cdot c(F) + O(1) \cdot \sigma(T) \\ \leq & O(1) \cdot \text{OPT} + O(1) \cdot \sigma(T) \\ & \quad \text{(since } c(F) \leq 2z(gst) \text{ and } z(gst) \leq \text{OPT}/k) \\ \leq & O(1) \cdot \text{OPT}. \end{array}$$

A similar analysis for the solution subgraphs $H(1)$, $H(2)$, $\ldots, H(7)$ shows that each has cost $\leq O(1) \cdot$ OPT.

Thus our claim for the cost of the solution subgraph H^* follows: $c(H^*) = O(\log r_{max}) \cdot$ OPT.

Finally, let us verify that H^* satisfies the connectivity requirements. Consider any pair of terminals i, j and their requirement $r_{i,j}$. Assume that $r_{i,j} \geq 8$ (otherwise, we are done by a similar but simpler analysis). Focus on an iteration of the algorithm that fixes the parameter k such that $k \leq r_{i,j} \leq 1.5k$. In that iteration, the requirements graph R has the edge $\{i, j\}$, hence, both i, j must be contained in one of the terminal sets $D_1, \ldots, D_q$, say D_1. Now, consider the sub-instance $\Pi'_1(k)$ and its solution subgraph $H'_1(k)$ and note that $H'_1(k)$ must have $\geq r_{i,j}$ openly disjoint i, j-paths because both r'_i and r'_j are $\geq r_{i,j}$ (here, r'_i and r'_j denote the requirements of i and j in $\Pi'_1(k)$) Thus, H^* has $\geq r_{i,j}$ openly disjoint i, j-paths.

This completes the proof of Theorem 4. ☐

Acknowledgments. We thank Bill Cunningham, Michel Goemans, Balaji Raghavachari, Ram Ravi, and Santosh Vempala for useful discussions over the years.

6. REFERENCES

[1] A.Agrawal, P.Klein and R.Ravi, "When trees collide : An approximation algorithm for the generalized Steiner problem on networks," *SIAM Journal on Computing*, **24**, 445–456, 1995. Preliminary version in *Proc. ACM STOC*, 1991.

[2] S.Arora, "Polynomial-time approximation schemes for Euclidean TSP and other geometric problems," *Journal of the ACM*, **45**, 753–782, 1998.

[3] M.Bern and P.Plassmann, "The Steiner problem with edge lengths 1 and 2," *Information Processing Letters*, **32**, 171–176, 1989.

[4] D.Bienstock, E.Brickell and C.Monma, "On the structure of minimum-weight k-connected spanning networks," *SIAM J. Discrete Math.*, **3**, 320–329, 1990.

[5] J.Cheriyan, T.Jordán and Z.Nutov, "On rooted node-connectivity problems," *Algorithmica*, **30**, 353–375, 2001.

[6] J.Cheriyan, S.Vempala and A.Vetta, "Approximation algorithms for minimum-cost k-vertex connected subgraphs," *Proc. 34th ACM STOC*, New York, 306–312, 2002.

[7] J.Cheriyan, S.Vempala and A.Vetta, "An approximation algorithm for the minimum-cost k-vertex connected subgraph," *SIAM Journal on Computing*, **32**, 1050–1055, 2003.

[8] N.Christofides, "Worst case analysis of a new heuristic for the traveling salesman problem," Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, 1976.

[9] A.Czumaj and A.Lingas, "On approximability of the minimum-cost k-connected spanning subgraph problem," *Proc. 10th Ann. ACM-SIAM Symposium on Discrete Algorithms*, 281–290, 1999.

[10] A.Czumaj, A.Lingas and H.Zhao, "Polynomial-time approximation schemes for the Euclidean survivable network design problem," *Proc. 29th ICALP*, LNCS **2380**, 973–984, 2002.

[11] G.B.Dantzig, L.R.Ford and D.R.Fulkerson, "Solution of a large-scale traveling-salesman problem," *Operations Research* **2**, 393–410, 1954.

[12] A.Frank, "Connectivity augmentation problems in network design," in *Mathematical Programming: State of the Art 1994*, (Eds. J. R. Birge and K. G. Murty), The University of Michigan, Ann Arbor, MI, 34–63, 1994.

[13] G.L.Frederickson and J.Ja'Ja', "On the relationship between the biconnectivity augmentation and traveling salesman problems," *Theor. Comp. Sci.* **19**, 189–201, 1982.

[14] M.Goemans and D.J.Bertsimas, "Survivable networks, linear programming relaxations and the parsimonious property," *Mathematical Programming*, **60**, 145–166, 1993.

[15] M.Goemans, A.Goldberg, S.Plotkin, D.Shmoys, E.Tardos and D.Williamson, "Improved approximation algorithms for network design problems," *Proc. 5th Ann. ACM-SIAM Symposium on Discrete Algorithms*, 223–232, 1994.

[16] M.Goemans and D.Williamson, "A general approximation technique for constrained forest problems," *SIAM Journal on Computing*, **24**, 296–317, 1995.

[17] K.Jain, "A factor 2 approximation algorithm for the generalized Steiner network problem," *Combinatorica*, **21**(1), 39–60, 2001. Preliminary version in *Proc. 39th IEEE FOCS*, 1998.

[18] S.Khuller, "Approximation algorithms for finding highly connected subgraphs," in *Approximation algorithms for NP-hard problems*, Ed. D.S.Hochbaum, PWS publishing co., Boston, 1996.

[19] S.Khuller and B.Raghavachari, "Improved approximation algorithms for uniform connectivity problems," *Journal of Algorithms* **21**, 434–450, 1996.

[20] G.Kortsarz, R.Krauthgamer and J.R.Lee, "Hardness of approximation for vertex-connectivity network design problems," *SIAM J. Computing* **33**, 704–720, 2004.

[21] G.Kortsarz and Z.Nutov, "Approximating k-node connected subgraphs via critical graphs," *Proc. 36th ACM STOC*, June 2004.

[22] G.Kortsarz and Z.Nutov, "Approximating node connectivity problems via set covers," *Algorithmica* **37**, 75–92, 2003. Preliminary version in *APPROX, Approximation algorithms for combinatorial optimization*, Springer, LNCS **1913**, 194–205, 2000.

[23] L.Lovász, *Combinatorial Problems and Exercises*, North-Holland, Amsterdam, and Akadémiai Kiadó, Budapest, 1979.

[24] C.L.Monma, B.S.Munson and W.R.Pulleyblank, "Minimum-weight two-connected spanning networks," *Mathematical Programming* **46**, 153–171, 1990.

[25] C.L.Monma and D.F.Shallcross, "Methods for designing communication networks with certain two-connectivity survivability constraints," *Operations Research* **37**, 531–541, 1989.

[26] J.S.B.Mitchell, "Guillotine subdivisions approximate polygonal subdivisions: a simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems," *SIAM J. Computing* **28**, 1298–1309, 1999.

[27] M.Stoer, *Design of Survivable Networks*, *Lecture Notes in Mathematics* **1531**, Springer-Verlag, Berlin, 1992.

[28] L.Trevisan, "When Hamming meets Euclid: the approximability of geometric TSP and MST," *SIAM Journal on Computing*, **30**, 475–485, 2001.

[29] V.V.Vazirani, *Approximation Algorithms*, Springer-Verlag, Berlin, 2001.

[30] D.Williamson, M.Goemans, M.Mihail and V.Vazirani, "A primal-dual approximation algorithm for generalized Steiner network problems," *Combinatorica* **15**, 435–454, 1995.