Online Node-weighted Steiner Tree and Related Problems

Joseph (Seffi) Naor  
Technion  
Haifa, Israel.  
Email: naor@cs.technion.ac.il

Debmalya Panigrahi  
Massachusetts Institute of Technology  
Cambridge, USA.  
Email: debmalya@mit.edu

Mohit Singh  
McGill University  
Montreal, Canada.  
Email: mohit@cs.mcgill.ca

Abstract—We obtain the first online algorithms for the node-weighted Steiner tree, Steiner forest and group Steiner tree problems that achieve a poly-logarithmic competitive ratio. Our algorithm for the Steiner tree problem runs in polynomial time, while those for the other two problems take quasi-polynomial time. We can view our algorithms as online LP rounding algorithms in the framework of Buchbinder and Naor; however, while the natural LP formulation of these problems do lead to fractional algorithms with a poly-logarithmic competitive ratio, we are unable to round these LPs online without losing a polynomial factor. Therefore, we design new LP formulations for these problems drawing on a combination of paradigms such as spider decompositions, low-depth Steiner trees, generalized group Steiner problems, etc., and use the additional structure provided by these LPs to round the more sophisticated LPs losing only a poly-logarithmic factor in the competitive ratio. As further applications of our techniques, we also design polynomial-time online algorithms with poly-logarithmic competitive ratios for two fundamental network design problems: the group Steiner forest problem (thereby resolving an open question raised by Chekuri et al) and the single source $\ell$-vertex connectivity problem (which complements similar results for the corresponding edge-connectivity problem due to Gupta et al).

1. INTRODUCTION

Network design problems, where the goal is to select a minimum cost subgraph of a given graph satisfying a given set of connectivity constraints, have played a crucial role in recent developments of many algorithmic paradigms. Perhaps the most well-known problem in this suite is the Steiner tree problem, in which the selected subgraph must connect a subset $T = \{t_i : 1 \leq i \leq k\}$ of designated vertices called terminals. In this paper, we consider the node-weighted (NW) version of this problem (where both edges and vertices have costs) in the classical online model (for the online model, see e.g. [6]), where the input graph $G = (V, E)$ is known in advance, but the terminals arrive online. The algorithm needs to ensure that at any stage of the online process, the subgraph selected thus far connects the terminals that have already arrived. We give the first algorithm for this problem with poly-logarithmic competitive ratio.2

Node weights are often used to model various practical scenarios such as the equipment cost at nodes of a real network, the load on network switches and routers [23], the latency and cost of recovery from power outages in electrical networks [21], etc. Further, from a theoretical perspective, node weights serve to unify edge-weighted network design problems and other classical covering problems such as set cover, facility location, etc. In fact, the NW Steiner tree problem, besides being a classical network design problem itself, also unifies (and generalizes) two fundamental optimization problems: the set cover problem by rounding (online) a fractional solution obtained from an LP-based primal-dual algorithm with multiplicative updates (for a comprehensive survey on this technique, see [7]); this algorithm has poly-logarithmic competitive ratio.3

- Alon et al [3] gave an algorithm for the online set cover problem by rounding (online) a fractional solution obtained from an LP-based primal-dual algorithm with multiplicative updates (for a comprehensive survey on this technique, see [7]); this algorithm has poly-logarithmic competitive ratio.

- Imase and Waxman [24] showed that the natural greedy algorithm has a logarithmic competitive ratio for the online EW Steiner tree problem.4

Part of the challenge in generalizing the above results lies in the contrasting techniques used to obtain them. We can easily rule out the greedy algorithm for the

2For a minimization problem, the competitive ratio of an online algorithm is the maximum ratio between the algorithmic solution and the offline optimal solution over all input sequences.

3Given a collection of subsets of a universe with respective costs, the set cover problem asks for a minimum cost sub-collection such that for every element of the universe, at least one subset containing it is in the sub-collection.

4LP stands for linear programming.
online NW Steiner tree problem (see Figure 1(a) for an input instance on which the greedy algorithm has a polynomial competitive ratio). On the other hand, for the LP-based approach, no online rounding technique is known for the standard LP formulation even in the EW case.

A key to understanding NW Steiner tree instances are spider decompositions, which were introduced by Klein and Ravi [26] for the offline version of the problem. In the online problem, it is a substantial challenge to even define spiders because of the dynamically changing set of terminals. We overcome this obstacle, and our key technical contribution in solving the online NW Steiner tree problem is two-fold:

- We prove a surprising structural property of NW Steiner trees showing that if we are ready to settle for a logarithmic loss, then the cost on very few vertices (and no edge) needs to be shared between terminals. This is in sharp contrast to the usual notion that the main challenge in the Steiner tree problem is in choosing between cheap edges/vertices that a few terminals pay for and expensive edges/vertices that many terminals share the cost on.
- The above property substantially simplifies the structure of a spider decomposition. This lets us write a (somewhat sophisticated) LP for the NW Steiner tree problem that unifies the set cover and EW Steiner tree problems, and sheds new light on the structure of the latter.

We then observe that this LP is identical to a non-metric facility location problem, for which Alon et al [2] gave an online algorithm with poly-logarithmic competitive ratio. Perhaps surprisingly, our algorithm exactly yields the greedy algorithm for EW Steiner tree [24] and the primal-dual algorithm for set cover [2], two algorithms that do not share any apparent similarity, when specialized to their respective instances.

Two generalizations of the Steiner tree problem that have also been extensively studied are:

- The Steiner forest problem, in which the subgraph must connect each pair \( (s_i, t_i) \) in a designated set of vertex pairs \( T = \{ (s_i, t_i) : 1 \leq i \leq k \} \) called terminal pairs.
- The group Steiner tree problem, in which the subgraph must connect the root vertex \( r \) to at least one vertex from every set \( T_i \) in a designated collection of vertex subsets \( T = \{ T_i : 1 \leq i \leq k \} \) called terminal groups.

While online algorithms with poly-logarithmic competitive ratios were known for the EW version of both these problems (see [4], [5] and [2] respectively), no non-trivial competitive ratio was known for the NW version. We give the first online algorithms for these problems with a poly-logarithmic competitive ratio. In fact, we give an algorithm for a somewhat more general problem (called the group Steiner forest problem) that unifies the above two problems. As further applications of our techniques, we also obtain polynomial-time online algorithms for the EW group Steiner forest problem and the EW single-source \( t \)-vertex connectivity problem. We formally define our problems and state our results next.

### 1.1. Our Results

We start by formally defining the online NW Steiner tree problem. Throughout the paper, for a graph \( G = (V, E) \), let \( |V| = n \) and \( |E| = m \). For a set cover instance, \( n \) and \( m \) respectively denote the number of elements and the number of sets.

**The Online Node-Weighted Steiner Tree Problem.** We are given (offline) an undirected graph \( G = (V, E) \), with cost \( c_e \) for edge \( e \in E \), and cost \( c_v \) for vertex \( v \in V \). A sequence of vertices (called terminals) \( T = (t_1, t_2, \ldots, t_k) \) \( (t_i \in V \) for \( 1 \leq i \leq k \) appear online; the algorithm needs to maintain a subgraph \( H \) of \( G \) that connects all the terminals, while minimizing the total cost of vertices and edges in \( H \). Our main result is the following theorem.

**Theorem 1.** There is a polynomial-time randomized online algorithm for the node-weighted Steiner tree problem with a competitive ratio of \( O(\log n \log^2 k) \).

We note that there is a lower bound of \( \Omega(\log n \log k) \) on the competitive ratio of this problem. This follows from a recent lower bound of \( \Omega(\log n \log n) \) on the competitive ratio of any randomized polynomial time algorithm for the online set cover problem, under the \( BPP \neq NP \) assumption [27].

We now define the group Steiner forest problem, which unifies (and generalizes) the Steiner forest problem and the group Steiner tree problem.

**The Online Node-weighted Group Steiner Forest Problem.** We are given (offline) an undirected graph \( G = (V, E) \) with cost \( c_e \) for edge \( e \in E \), and cost \( c_v \) for vertex \( v \in V \). A sequence of pairs of vertex subsets (called terminal group pairs) \( \mathcal{T} = \{ (S_1, T_1), (S_2, T_2), \ldots, (S_k, T_k) \} \) \( (S_i, T_i) \subseteq V \) for \( 1 \leq i \leq k \) appear online; the algorithm needs to maintain a subgraph \( H \) of \( G \) that connects at least one pair of vertices \( s_i \in S_i, t_i \in T_i \) for each terminal group pair \( (S_i, T_i) \), while minimizing the total cost of vertices and edges in \( H \). (Note that the Steiner forest and group Steiner tree
problems are special cases with $|S_i| = |T_i| = 1$, $\forall i$ and $S_i = \{v\}$, $\forall i$ respectively.) We obtain an online algorithm for this problem with poly-logarithmic competitive ratio.

**Theorem 2.** There is a quasi-polynomial-time randomized online algorithm for the node-weighted group Steiner forest problem with a competitive ratio of $O(polylog(n,k))$.

### Online Edge-weighted Network Design Problems.

Our techniques also lead to new results in online EW network design. First, we give a polynomial-time online algorithm for the EW group Steiner forest problem, thus resolving an open question raised by Chekuri et al [11] in the affirmative.

**Theorem 3.** There is polynomial-time randomized online algorithm$^7$ for the edge-weighted group Steiner forest problem with a competitive ratio of $O(polylog(n,k))$.

Next, for the EW single-source $\ell$-vertex connectivity problem, where the goal is to find a minimum cost subset of edges such that each terminal has at least $\ell$ vertex-disjoint paths to a fixed root vertex, we obtain the following theorem.

**Theorem 4.** There is a polynomial-time deterministic online algorithm for the edge-weighted single-source $\ell$-vertex connectivity problem with a (bicriteria)$^8$ competitive ratio of $O\left(\frac{\ell \log k}{\epsilon^2}, 2 + \epsilon\right)$ for any $\epsilon > 0$.

This theorem complements the results of Gupta et al [22] for the corresponding online edge-connectivity problem.

1.2. Our Techniques

As noted earlier, the NW Steiner tree problem generalizes both the EW Steiner tree problem and the set cover problem; therefore, a natural approach is to unify (and generalize) the online algorithms for these problems. We first discuss the challenges faced by these approaches. For the online EW Steiner tree problem, there are mainly two approaches. The first one is the greedy algorithm (each terminal connects via a shortest path to the previous terminals) which is known to be $O(\log n)$-competitive. Unfortunately, for the NW version, the greedy algorithm has a polynomial competitive ratio (see Figure 1(a)).

The second approach is based on probabilistic tree embeddings [17], which have been successfully used by Gupta et al [22] even for higher connectivity requirements in EW online settings. However, such tree embeddings do not exist if vertices have costs, ruling out such an approach.

The online set cover problem has an $O(\log m \log n)$ competitive algorithm [3] which works by first solving online the standard LP within an $O(\log m)$ factor, and then adapting the randomized rounding method for set cover to work online, losing another factor of $O(\log n)$. Using the methods of [2] for online covering of cuts in a graph, it is not hard to show that a fractional

---

$^5$The exact competitive ratio in Theorem 2 is $O(\log k \log^3 n)$, while that in Theorem 3 is $O(\log^2 n \log k)$.

$^6$An algorithm is said to be quasi-polynomial-time if its time complexity is $O(k^{\text{polylog}})$, where $k$ is the input to the algorithm.

$^7$In fact, our (online) algorithm is somewhat simpler than the previously known offline algorithm for this problem [11], though the previous algorithm has a slightly better approximation ratio of $O(\log^2 n \log^2 k)$ (versus $O(\log^3 n \log k)$ for our algorithm).

$^8$A bi-criteria competitive ratio of $(a,b)$ for an $\ell$-connectivity problem implies that the solution produced by the online algorithm achieves a connectivity of $\ell/b$ and is at most a factor of $a$ more expensive than the optimal offline solution for connectivity $\ell$. 

Figure 1. (a) A counter-example to a greedy algorithm for the online node-weighted Steiner tree problem. All the curved edges have cost $1 - \epsilon$ and the straight edges have cost $0$. Vertex $v$ has cost $1$; all other vertices are terminals and have cost $0$. In this example, the vertices $\{r, t_1, t_2, \ldots, t_k\}$ appear as terminals. Since each terminal has a private path of cost $1 - \epsilon$ to $r$, the greedy algorithm selects these private paths with total cost $(1 - \epsilon)k$, whereas the optimal solution chooses the paths through vertex $v$ and has cost $1$. Choosing $\epsilon$ to be an arbitrarily small positive constant leads to a lower bound of $k$ on the competitive ratio of this algorithm. (b) An example exhibiting the difficulty of online rounding of the natural LP relaxation of NW Steiner tree. If each edge and vertex has a value of $1/\sqrt{n}$ in the fractional solution, then an independent rounding of the edges and vertices does not produce a feasible solution. On the other hand, since the value on an edge or vertex accumulates over multiple rounds, dependent rounding may produce an integer solution that is polynomially more expensive than the fractional solution.
solution to the standard LP formulation of the NW Steiner tree problem can be computed online and has a poly-logarithmic competitive ratio. Now, rounding the fractional solution online (as in set cover) is the natural approach to obtaining an online algorithm with a poly-logarithmic competitive factor. However, this LP appears to be too weak to allow for this kind of rounding without losing a polynomial factor in the competitive ratio (see Figure 1(b) for an illustrative example). Moreover, even for the EW Steiner tree problem, we do not know how to round a fractional solution to this LP. In fact, one of our main contributions is developing an approach that unifies two seemingly different algorithms: the greedy algorithm for the EW Steiner tree problem and the online LP rounding based approach to the set cover problem. We describe below our new approach.

Observe that one can view a solution to the online Steiner tree problem as a collection of paths, one from each terminal to another terminal that appeared earlier in the online sequence. If each terminal could afford to pay for its entire path (to the previous terminal), then a greedy algorithm suffices. In fact, for the EW version this is indeed the case, and this property is crucial for the analysis of the greedy algorithm in the online setting. However, as indicated earlier, the example in Figure 1(a) asserts that for the NW version this property is not true, i.e., terminals must necessarily share the cost of these paths in order to obtain a poly-logarithmic competitive ratio.

A natural next step is bounding the extent of the cost sharing among the terminals. For example, in Figure 1(a), terminals $t_1, t_2, \ldots, t_k$ only need to share the cost on the solitary vertex $v$ on their paths to terminal $r$. Our key lemma, somewhat surprisingly, generalizes this to show that if we are ready to sacrifice a factor of $O(\log k)$ in the cost, then the cost sharing among terminals can be restricted to a single vertex on every path.

**Lemma 1.** Let $G = (V, E)$ be an undirected graph with vertex and edge costs $c_v, c_e$ respectively. Suppose $T \subseteq V$ is a set of $k$ terminal vertices. Then, for any ordering of the terminals $t_1, t_2, \ldots, t_k$, and for any subgraph $G_T$ of $G$ connecting all the terminals, there exists a set of paths $P_1, P_2, \ldots, P_k$ and a corresponding set of vertices $v_2, v_3, \ldots, v_k$ such that

- $P_i$ is a path from terminal $t_i$ to another terminal $t_j$ which is earlier in the order, i.e., $j < i$.
- $v_k$ is on path $P_i$ and is also contained in $G_T$, and
- $\sum_{i=2}^{k} (c(P_i) - c_{v_k} \leq O(\log k) \cdot c(G_T))$,

where $c(P_i)$ is the sum of costs of vertices and edges on $P_i$, and $c(G_T)$ is the sum of costs of vertices and edges in $G_T$.

Our main challenge, then, is to select vertex $v_i$ and path $P_i$ for each terminal $t_i$ that arrives online. In fact, the crux of this selection is in the choice of $v_i$; once $v_i$ is chosen, we can greedily add the cheapest path from $t_i$ to $v_i$, as well as the one from $v_i$ to any $t_j, j < i$, to obtain path $P_i$. Note that since terminal $t_i$ can exclusively pay for path $P_i$, except for vertex $v_i$, a greedy choice of $P_i$, given $v_i$, is optimal. This observation lets us encode the Steiner tree problem as a non-metric facility location problem for which an $O(\log n \log k)$-competitive algorithm was given by Alon et al [2]. This ultimately leads to Theorem 1.

**The Online Node-weighted Group Steiner Forest Problem.** We now turn our attention to the online node-weighted group Steiner forest problem. As with the Steiner tree problem, the methods of [2] for online covering of cuts in a graph can be used to obtain a fractional solution with a poly-logarithmic competitive ratio for the standard LP formulation of this problem. However, this LP seems too weak to allow online rounding; so one might hope for a strengthening of the LP similar to Steiner tree. Unfortunately, for the group Steiner forest problem, the cost sharing property in Lemma 1 does not generalize.

Instead we give a different approach for strengthening the LP. We first prove a structural result that there is a near-optimal feasible solution in the form of a forest such that every tree in the forest has small depth.\(^9\) The technical lemma behind this claim is a generalization

\(^9\)Edges in the low-depth trees represent paths in the original trees.
of a similar result by Robins and Zelikovsky [32] on EW graphs. In view of this structural result, we reduce this problem on general graphs to trees in a natural manner. The size of the tree instances we create are bounded by $O(n^h)$ where $h$ is the height of the tree. The structural lemma ensures that $h = \log k$ in our case, thereby guaranteeing that the time complexity of our algorithm stays quasi-polynomial.\(^\text{10}\)

We complement the above reduction by giving an online algorithm for the group Steiner forest problem on a tree with poly-logarithmic competitive ratio.

**Theorem 5.** There is an $O(h\log^3 n \log k)$-competitive randomized online algorithm for the group Steiner forest problem on trees of depth $h$ with both edge and vertex costs.

Combining all the pieces together, we obtain Theorem 2.

**Online Edge-weighted Network Design Problems:**

An another application of Theorem 5, we use randomized low-distortion embeddings of graph into low-depth trees (due to Fakcharoenphol et al [17]) to give an online polynomial-time algorithm for the group Steiner forest problem in EW graphs thereby proving Theorem 3. On the other hand, the proof of Theorem 4 uses a combination of two sets of techniques: our decomposition of spiders into paths in terminal arrival order (Lemma 1), and a generalization of spider decompositions to higher connectivity developed by Chuzhoy and Khanna [15] (and simplified later by Chekuri and Korula [14]) for the offline version of the problem.

1.3. Related Work

Klein and Ravi [26] introduced the notion of spider decomposition to give the first (optimal\(^\text{11}\)) $O(\log k)$-approximation algorithm for the (offline) NW Steiner tree problem (and for some generalizations including the NW Steiner forest problem). In subsequent work, algorithms with better approximation ratios have been developed for various special cases (see e.g. [16], [29], [34]) and for higher (and more general) connectivity requirements [31], [28], [15].

A different network model was considered by Guha et al [21] and Moss and Rabani [30] where each node has a cost and a profit, and the goal is to satisfy the desired connectivity requirements while minimizing cost and maximizing profit. Buy-at-bulk network design problems on NW graphs have also been studied previously [13]. Further, routing problems in node capacitated graphs have also been studied extensively (see e.g. GargVY04, ChekuriKS05, HajiaghayiKRL07, FeigeHL08), though these problems do not typically have a minimum cost objective.

Much of prior research in network design problems has concentrated on EW versions. A series of algorithms (e.g. [33], [32]) for (offline) EW Steiner tree has led to the current best approximation factor of 1.39 [8]. For the EW Steiner forest problem, Agrawal et al [1] (and then Goemans and Williamson [19]) gave a primal-dual algorithm with an approximation factor of 2 (which was matched by Jain [25] for the generalized Steiner forest problem). For the EW group Steiner tree and EW group Steiner forest problems, Garg et al [18] and Chekuri et al [11] gave the first algorithms to achieve a poly-logarithmic approximation ratio (see [9], [10], [12] for later improvements in group Steiner tree).

In the online model, Imase and Waxman [24] and Awerbuch et al [4] respectively showed that the greedy algorithm has a competitive ratio of $O(\log n)$ and $\Theta(\log^2 n)$ for the EW Steiner tree and the EW Steiner forest problem (see [5] for a subsequent $\Theta(\log n)$-competitive algorithm for the EW Steiner forest problem). Later, Alon et al [2] used an online primal-dual technique originally developed for set cover to obtain the first poly-logarithmic competitive ratio for the EW group Steiner tree problem. The design of a poly-logarithmic competitive online algorithm for the EW group Steiner forest problem was an open question raised in [11]; we settle this question in the affirmative.

2. Online Node-weighted Steiner Tree

In this section, we prove Theorem 1 for which our first goal is to prove Lemma 1, the key tool in our algorithm. To prove this lemma, we need to introduce the technique of spider decomposition of trees due to Klein and Ravi [26].

**Definition 1.** A spider is a connected graph containing at least three vertices, where at most one vertex has degree greater than two. Each vertex that has degree equal to one is called a foot, while the unique vertex that has degree greater than two is called the head. If no vertex has degree greater than two, then any of the vertices with degree equal to two can be called the head. A head-to-foot path is called a leg of the spider.

Klein and Ravi [26] defined the notion of a spider decomposition of a tree and proved its existence.
Lemma 2 (Klein-Ravi [26]). Any tree $R$ contains a set of vertex-disjoint spiders such that the feet of the spiders are exactly the leaves of the tree. This set is called a spider decomposition.

![Spider decomposition](image)

Figure 3. This figure shows a covering spider decomposition of a tree. The leaves are ordered as $(t_1, t_2, t_3, t_4, t_5)$. The spiders in the top right corner form a spider decomposition of the tree. The red arrows indicate the paths used by $t_2$ to connect to $t_1$, $t_2$ to $t_2$ and $t_3$ to $t_2$, and $t_4$ to $t_5$ in the proof of Lemma 1. In the second recursive level, there are only two terminals $t_1$ and $t_5$, which were the two earliest terminals in their respective spiders. Thus, there is a single spider connecting them, and this spider has a path from $t_4$ to $t_1$. In general, instead of two recursive levels, we might have log $k$ recursive levels.

We extend this lemma to produce a recursive spider decomposition $\mathcal{S}$ of any tree $R$. Suppose $\mathcal{L} = \ell_1, \ell_2, \ldots, \ell_k$ is an arbitrary ordering of the leaves of tree $R$. A covering spider decomposition (see Figure 3 for an example) of $R$ with respect to the ordering $\mathcal{L}$ is a sequence of sets of spiders $\mathcal{S}_1, \mathcal{S}_2, \ldots$ with the following properties:

- The spiders in any set $\mathcal{S}_i$ are node-disjoint.
- $\mathcal{S}_i$ is a spider decomposition of $R$, i.e., the feet of the spiders in $\mathcal{S}_i$ are the leaves of $R$.
- Let $\mathcal{L}_i = \{s_{i1}, s_{i2}, \ldots, s_{ir_i}\}$. Now, let the leaves of $R$ that are feet of spider $s_{ij}$ be $L_{ij}$; further let $\ell_{ij}$ be the first among these leaves in the ordering $\mathcal{L}$. Then, the feet of the spiders in $\mathcal{S}_{i+1}$ are exactly the leaves $\{\ell_{ij} : 1 \leq j \leq r_i\}$.

Before showing that such a recursive decomposition of spiders exists for any tree, let us show that its existence implies Lemma 1. Recall that in Lemma 1, $G_T$ is a connected subgraph of $G$ containing all the terminals in $T$. Let $R$ be a spanning tree of $G_T$. We can also assume that all terminals in $T$ are leaves of $R$\(^{12}\).

Now, for each terminal $t_i$, the path $P_i$ and the vertex $v_i$ on it (as in Lemma 1) are defined as follows.

Let $j_i$ be the maximum index $j$ such that terminal $t_i$ is a foot in a spider $s \in \mathcal{S}_j$. Let $T_j$ be the ordering of the terminals that are feet of spider $s$ with respect to arrival order. Then, we define the path $P_i$ as the path from $t_i$ to the terminal immediately before $t_i$ in the sequence $T_j$. Also, $v_i$ is defined as the head of spider $s$. The following property is a direct consequence of this definition.

Lemma 3. The sum of costs $c(P_i) - c_{v_i}$ for all terminals $t_i$ having $j_i = j$ for a fixed $j$ is at most $2c(R)$.

Proof: The proof follows by observing that each leg of a spider $s \in \mathcal{S}_j$ appears in path $P_i$ for at most two terminals $t_i$ having $j_i = j$.

The next lemma follows from the fact that each spider must contain at least two feet.

Lemma 4. The number of sets of spiders in a covering spider decomposition of a tree containing $O(k)$ vertices is $O(\log k)$, irrespective of the ordering of the leaves.

The above two lemmas immediately imply Lemma 1.

Finally, we need to show that any tree has a covering spider decomposition with respect to any ordering of the leaves. We give a recursive procedure for constructing such a decomposition. First, we use Lemma 2 to produce a spider decomposition $\mathcal{S}_1$. Then, we delete all the legs of each spider in $\mathcal{S}_1$, except the leg that ends at the leaf that appears earliest in the ordering among the leaves in $s$. We now recursively construct the spider decompositions $\mathcal{S}_2, \mathcal{S}_3, \ldots$ in the remaining tree. This completes the proof of Lemma 1.

It is interesting to note that Lemma 1 implies that no cost sharing is necessary in the edge-weighted case. The next corollary formalizes this claim.

Corollary 1. Let $G = (V,E)$ be an undirected graph with edge costs only. Suppose $T \subseteq V$ is a set of $k$ terminal vertices. Then, for any ordering of the terminals $t_1, t_2, \ldots, t_k$, and for any subgraph $G_T$ of $G$ connecting all the terminals, there exists a set of paths $P_2, P_3, \ldots, P_k$ such that:

- $P_i$ has endpoints $t_i$ and $t_j$ for some $j < i$,
- $\sum_{i=2}^{k} c(P_i) \leq O(\log n)c(G_T)$,
- $c(P_i)$ is the sum of costs of edges on $P_i$, and $c(G_T)$ is the sum of costs of edges in $G_T$.

It follows from Corollary 1 that the greedy algorithm for the online edge-weighted Steiner tree problem is $O(\log n)$-competitive, providing an alternative proof for this well known result.

Our goal now is to select vertex $v_i$ and path $P_i$ for each terminal $t_i$; in fact, as observed earlier, selecting
for the facility location problem with competitive ratio.

Algorithm of Alon to facilities. Using Lemma 1, we conclude that the
observation allows us to encode the Steiner tree problem

Consider the set of terminals

In the online version of the problem,

Now, we claim that the integer program in Figure 4

Recall that the group Steiner forest problem is defined
as follows. Let $G = (V, E)$ be an undirected graph and $\mathcal{F} = \{(S_1, T_1), (S_2, T_2), \ldots, (S_k, T_k)\}$ be $k$ pairs of
subsets of vertices called terminal group pairs. We need
to find a minimum cost subgraph $H$ such that for each
terminal group pair $(S_i, T_i)$, $H$ connects at least one pair
of vertices $s_i \in S_i, t_i \in T_i$.

In this section, we give an online algorithm for this
problem when the input graph is a tree, and prove
Theorem 5. Our algorithm has two stages. In the first
stage (details omitted due to lack of space), we use
an online primal-dual algorithm for generalized cut
problems due to Alon et al [2] to obtain a fractional solution for our problem satisfying the next lemma.

Lemma 5. The fractional solution for the online node-weighted group Steiner forest problem has cost at most $O(\alpha \cdot \log n)$, where $\alpha$ is the cost of an (offline) optimum integer solution.

In the second stage of the algorithm, we give a
randomized algorithm for rounding the fractional
solution to an integer solution. The basic idea is to run a
rounding technique for the group Steiner tree problem
on a tree due to Garg et al [18] (whose online version
was given by Alon et al [2]) for every subtree of the
rooted input tree. We will show that the integer solution
connects at least one pair of vertices from each terminal
group pair $(S_i, T_i)$ with probability $Q(1/\log^2 n)$.

Moreover, the expected cost of the integer solution is
$O(h \alpha)$, where $\alpha$ is the (offline) optimum cost and $h$ is
the height of the input tree. We run $O(\log^2 n \log k)$
parallel instantiations of this rounding technique; using
standard analysis, we then conclude that all terminal
group pairs are satisfied with probability at least $1 - 1/k$.

This allows us to add the cheapest path from a vertex in
$S_i$ to a vertex in $T_i$ for an unsatisfied group pair $(S_i, T_i)$;
the expected overhead because of this step is $O(\alpha)$ since
the cheapest path has cost at most $\alpha$.

Suppose the new group pair in an online step is $(S_i, T_i)$. Our first step is to identify a collective flow of one between vertices in $S_i$ and $T_i$ that can be supported by the fractional solution. We decompose this flow into flow paths characterized by their endpoints $(s_{ij}, t_{ij}), (s_{ij}, t_{ij}), \ldots$, where $s_{ij} \in S_i, t_{ij} \in T_i$. Let $f^{(v)}_i(e)$ (resp., $f^{(v)}_i(u)$) denote the total flow routed through
edge $e$ (resp., vertex $u$) on flowpaths such that the

Figure 4. An integer linear program for the online node-weighted Steiner tree problem.

$v_i$ immediately selects the path $P_i$ as the cheapest path
from $t_i$ to $v_i$, and then from $v_i$ to some $t_j, j < i$. This
observation allows us to encode the Steiner tree problem
as an integer linear program in Figure 4.

In the linear program, $c^{(v)}_i$ is the sum of the costs of
the cheapest path from terminal $t_i$ to vertex $v$ and the
cheapest path from $v$ to any of previous terminals, i.e.
any $t_j, j < i$. Both of these costs do not include the
cost of $v$. The variable $x^{(v)}_i$ is an indicator variable for
the event $v = v_i$. The first constraint guarantees that for
each terminal $t_i$, we choose at least one vertex as $v_i$; the
second constraint guarantees that if a vertex $v$ is chosen
as $v_i$ by at least one terminal $t_i$, then we pay $c_v$ in the
objective value.

Now, we claim that the integer program in Figure 4
can be modeled as a (non-metric) facility location
problem. In an instance of the facility location problem,
we are given a set of possible facilities $F$ and a set
of clients $C$. We are also given a facility opening cost
c $c_f$ for each $f \in F$ and a client connection cost $c_{if}$
for each client $i \in C$ and $f \in F$. The goal is to open the
facilities and assign clients to open facilities such that the
sum of facility opening costs and connection costs is
minimized. In the online version of the problem,
the clients arrive online; when a client arrives, we can
either open a new facility and connect the client to it
or connect the client to a previously opened facility.

Alon et al [2] gave a randomized online algorithm
for the facility location problem with competitive ratio
$O(\log n \log k)$, where $|C| = k$ and $|F| = n$.

To model the integer program in Figure 4 as a facility
location problem, we give the following reduction.
Consider the set of terminals $t_i, 2 \leq i \leq k$, as clients
and the vertices $v \in V$ as facilities. The cost of opening
a facility $v$ is $c_v$, while the connection cost of serving
a client $t_i$ using facility $v$ is $c^{(v)}_i$. Then, the linear program
in Figure 4 asks for the cheapest assignment of clients
to facilities. Using Lemma 1, we conclude that the
algorithm of Alon et al [2], applied to our facility
location instance, yields an $O(\log n \log^2 k)$-competitive
algorithm for the online NW Steiner tree problem,
thereby proving Theorem 1.

3. ONLINE NODE-WEIGHTED GROUP STEINER
FOREST IN TREES

Recall that the group Steiner forest problem is defined
as follows. Let $G = (V, E)$ be an undirected graph and $\mathcal{F} = \{(S_1, T_1), (S_2, T_2), \ldots, (S_k, T_k)\}$ be $k$ pairs of
subsets of vertices called terminal group pairs. We need
to find a minimum cost subgraph $H$ such that for each
terminal group pair $(S_i, T_i)$, $H$ connects at least one pair
of vertices $s_i \in S_i, t_i \in T_i$.

In this section, we give an online algorithm for this
problem when the input graph is a tree, and prove
Theorem 5. Our algorithm has two stages. In the first
stage (details omitted due to lack of space), we use
an online primal-dual algorithm for generalized cut
problems due to Alon et al [2] to obtain a fractional solution for our problem satisfying the next lemma.

Lemma 5. The fractional solution for the online node-weighted group Steiner forest problem has cost at most $O(\alpha \cdot \log n)$, where $\alpha$ is the cost of an (offline) optimum integer solution.

In the second stage of the algorithm, we give a
randomized algorithm for rounding the fractional
solution to an integer solution. The basic idea is to run a
rounding technique for the group Steiner tree problem
on a tree due to Garg et al [18] (whose online version
was given by Alon et al [2]) for every subtree of the
rooted input tree. We will show that the integer solution
connects at least one pair of vertices from each terminal
group pair $(S_i, T_i)$ with probability $Q(1/\log^2 n)$.

Moreover, the expected cost of the integer solution is
$O(h \alpha)$, where $\alpha$ is the (offline) optimum cost and $h$ is
the height of the input tree. We run $O(\log^2 n \log k)$
parallel instantiations of this rounding technique; using
standard analysis, we then conclude that all terminal
group pairs are satisfied with probability at least $1 - 1/k$.

This allows us to add the cheapest path from a vertex in
$S_i$ to a vertex in $T_i$ for an unsatisfied group pair $(S_i, T_i)$;
the expected overhead because of this step is $O(\alpha)$ since
the cheapest path has cost at most $\alpha$.

Suppose the new group pair in an online step is
$(S_i, T_i)$. Our first step is to identify a collective flow of one between vertices in $S_i$ and $T_i$ that can be supported by the fractional solution. We decompose this flow into flow paths characterized by their endpoints
$(s_{ij}, t_{ij}), (s_{ij}, t_{ij}), \ldots$, where $s_{ij} \in S_i, t_{ij} \in T_i$. Let $f^{(v)}_i(e)$ (resp., $f^{(v)}_i(u)$) denote the total flow routed through
edge $e$ (resp., vertex $u$) on flowpaths such that the

\[
\begin{align*}
\min & \quad \sum_{v \in V} x^{(v)}_i + \sum_{v \in V} c_v y_v \\
\text{s.t.:} & \quad \sum_{v \in V} x^{(v)}_i \geq 1 \quad \forall 2 \leq i \leq k \\
& \quad x^{(v)}_i \leq y_v \quad \forall v \in V, 2 \leq i \leq k \\
& \quad x^{(v)}_i \in \{0, 1\} \quad \forall v \in V, 2 \leq i \leq k \\
& \quad y_v \in \{0, 1\} \quad \forall v \in V.
\end{align*}
\]
least common ancestor\textsuperscript{13} of $s_i, t_i$ in the input tree $R$ is vertex $v$. Here, $e$ is an edge and $u$ is a vertex in the subtree rooted at $v$ in $R$ (we denote this subtree $R_v$). For technical reasons, we double the value of $f_i(v)$. This lets us view the flow from $S_i$ to $T_i$ through $v$ in $R_v$ as a flow from $S_i$ to $v$ and a separate flow from $T_i$ to $v$ of the same value. Now, let $x_i^v(e) = \max_{j \leq i} \{(f_j^v(v), f_j^v(u))\}$ and $x_i^v(u) = \max_{j \leq i} \{f_j^v(u)\}$. Observe that $x_i^v(e)$ and $x_i^v(u)$ are monotonically decreasing as we move down from the root $v$ in $R_v$. Also, $x_i^v(e) \leq x_e$ and $x_i^v(u) \leq x_u$ (after online round $i$). Finally, note that the values of $x_i^v(e)$ and $x_i^v(u)$ are non-decreasing during the course of the online algorithm (i.e. with increase in $i$). This lets us apply the rounding algorithm of Alon et al [2] to the solution $x_i^v(e) / f_i(v)$ twice (if $f_i(v) > 0$) and then select the output of both instances with probability $f_i^v(v)$, and reject the output with probability $1 – f_i(v)$. We will now prove the next lemma using a technique similar to [2].

**Lemma 6.** For each group pair $(S_i, T_i)$, the randomized rounding procedure selects a path from some vertex in $S_i$ to some vertex in $T_i$ with probability $\Omega \left( \frac{1}{\log^2 n} \right)$. Further, the cost of the integer solution obtained by the randomized rounding procedure is $O(h \cdot \alpha)$.

In order to prove Lemma 6, we use the next lemma.

**Lemma 7.** For any $R_v$, a path is selected in the integer solution from $v$ to some vertex in $S_i$ and some vertex in $T_i$ with probability at least $\frac{f_i^v(v)}{\log^2 n}$. Moreover each edge $e$ (resp., vertex $u$) in $R_v$ is selected with probability at most $2x_i^v(e)$ (resp., $x_i^v(u)$).

**Proof:** Consider any $v$ such that $f_i^v(v) > 0$. Observe that the solution $x_i^v(e) / f_i^v(v)$ is a feasible solution to the group Steiner tree problem of connecting the group $S_i$ (resp., $T_i$) to root $v$. Alon et al’s analysis (in particular, Lemma 12 in [2]) implies that the rounding algorithm connects some vertex in $S_i$ to $v$ with probability at least $1 / \log n$ in the first instance, also some vertex in $T_i$ to $v$ with probability $1 / \log n$ in the second instance. Since, the two runs are independent and we select the output with probability $f_i^v(v)$, we conclude that a path is selected from some vertex in $S_i$ to $T_i$ through $v$ with probability at least $f_i^v(v) / \log^2 n$.

To bound the cost, observe that the rounding algorithm of Alon et al [2] in a single run selects any edge $e$ with probability at most $x_i^v(e) / f_i^v(v)$ and we conditionally select it with probability $f_i^v(v)$. This implies that over the two rounds, we select edge $e$ with probability at most $2 \cdot \frac{f_i^v(v)}{f_i^v(v)} \cdot f_i^v(v) = x_i^v(e)$. A similar argument proves the bound for each vertex $u$.

Using the above lemma, we now prove Lemma 6.

**Proof of Lemma 6:** For any group pair $(S_i, T_i)$, the probability that a path between some $s_i \in S_i$ and some $t_i \in T_i$ is selected is

$$1 - \prod_{v \in V} \left( 1 - \frac{\Omega(f_i^v(v))}{\log^2 n} \right) = \Omega \left( \frac{1}{\log^2 n} \right),$$

since $\sum_{v \in V} f_i^v(v) = 1$.

To bound the cost of the integer solution, note that each vertex $u$ or edge $e$ appears in at most $h$ subtrees $R_v$, and the integer solution obtained from each tree $R_v$ has cost at most $\sum_{v \in V} x_i^v(u) + \sum_{e \in E} x_i^v(e) \leq \sum_{v \in V} x_v + \sum_{e \in E} x_e \leq \alpha$.

Lemmas 5 and 6 immediately imply Theorem 5.

### 4. Online Group Steiner Forest in General Graphs

In this section, we use the online NW group Steiner forest algorithm on trees (from the previous section) to obtain algorithms for the corresponding NW and EW problems on general graphs.

#### 4.1. Online Node-weighted Group Steiner Forest

We first consider the NW version of this problem on general graphs. The following structural lemma about an offline optimal solution (which generalizes a similar lemma for the EW case due to Robins and Zelikovsky [32]) is key to our reduction of this problem to the corresponding problem on trees (proof omitted due to lack of space).

**Lemma 8.** Given any instance of NW group Steiner forest problem on a graph $G = (V, E)$ with terminal group pairs $\mathcal{F} = \{(S_1, T_1), (S_2, T_2), \ldots, (S_k, T_k)\}$ ($S_i, T_i \subseteq V$ for $1 \leq i \leq k$), there exists another instance of the NW group Steiner forest problem on a graph $G' = (V', E')$ where $V \subseteq V'$ with the same terminal group pairs $\mathcal{F} = \{(S_1, T_1), (S_2, T_2), \ldots, (S_k, T_k)\}$ such that

1. For any feasible solution $H$ for the the instance on graph $G$, there exists a feasible solution $H'$ for the instance on graph $G'$ such that $c(H') \leq c(H) \cdot \log n$.

2. For any feasible solution $H'$ for the instance on graph $G'$, there exists a feasible solution $H$ for the instance on graph $G$ such that $c(H) \leq c(H')$.\textsuperscript{13}
Moreover, there is an optimal solution $H'$ for $G'$ such that every tree in $H'$ has depth at most $\log n$.

Using Lemma 8, we construct a reduction from NW group Steiner forest on general graphs to trees, and show the next lemma.

**Lemma 9.** Given an instance of the group Steiner forest problem on a graph $G = (V, E)$ such that each tree of the optimal forest has depth at most $\log n$, there exists an instance of the group Steiner forest problem on a tree $T$ of size $O(n \log n)$ such that every feasible solution on $G$ corresponds to a feasible solution on $T$ of the same cost and vice-versa.

**Proof:** Given an instance of the NW group Steiner forest problem on a graph $G = (V, E)$ such that every tree has depth at most $\log n$ in the optimal solution, we construct the tree $R$ as follows. The tree $R = (V_R, E_R)$ has $\log n + 1$ levels indexed by $0, 1, 2, \ldots, \log n$. Level $i$ contains $n^i$ copies of each vertex in $V$, the cost of each copy of a vertex being its cost in $G$. To index these vertices, let us first arbitrarily index the vertices in $V$ as $\{v_1, v_2, \ldots, v_n\}$. Then, the vertices in level $i$ are denoted by $\{v_{p, j_1, j_2, \ldots, j_l}^i\}$ where each $1 \leq j_s \leq n$ and $1 \leq p \leq n$. The set of edges $E$ comprises the following sets of edges:

- For each $i \geq 0$, edges between $(v_{p, j_1, j_2, \ldots, j_l})$ and $(v_{q, j_1, j_2, \ldots, j_l})$ for each $1 \leq p, q \leq n$, of cost $d_{v_p v_q}$, i.e. the distance between vertices $v_p$ and $v_q$ in graph $G$.

For a terminal group pair $(S_i, T_i)$ in the NW group Steiner forest problem, we introduce a set $(S'_i, T'_i)$ where $S'_i$ contains all copies of the vertices in $S_i$ and $T'_i$ contains all copies of the vertices in $T_i$.

Consider any group Steiner forest $H$ in $G$. Without loss of generality $H$ is a forest. Root every tree $T$ of $H$ at any vertex, say $v_T$. Indeed, it is easy to see that there is a copy of this tree rooted at the unique copy of vertex $v_T$ at level 0. Observing this for every tree $T$, we obtain a solution $H'$ in the tree $R$ of cost equal to the cost of $H$.

Consider any group Steiner forest $H'$ on the tree $R$. Without loss of generality $H'$ is a forest. For any tree $T'$ in $H'$, there exists a subgraph $T$ in $G$ connecting exactly the same set of nodes connected by $T'$ and of no greater cost. This follows from the fact that every edge between a copy of node $u$ and $v$ in $R$ corresponds to a path between $u$ and $v$ in $G$. Hence, $H$ is a group Steiner forest.

Lemma 9 and Theorem 5 immediately lead to a proof of Theorem 2.

### 4.2. Online Edge-weighted Group Steiner Forest

We now give a polynomial-time algorithm for the online EW group Steiner forest problem and prove Theorem 3. The algorithm follows from a simple reduction to the EW group Steiner forest problem on a tree using small-depth low-distortion probabilistic tree embeddings [17].

**Lemma 10.** Given any instance of the online EW group Steiner forest problem on a graph $G = (V, E)$, there exists a distribution on trees $T = (V, E_T)$ such that

1. Every feasible solution $H$ on $G$ corresponds to a feasible solution $H'$ on $T$ such that $E_T[c(H')] \leq c(H) \log n$.

2. Every feasible solution $H'$ on any tree $T$ corresponds to a feasible solution $H$ on $G$ such that $c(H) \leq c(H')$.

Moreover, the height of any tree is $O(\log \Delta)$ where $\Delta$ is the diameter of $G$ and minimum length is one. Also, on any root to leaf path, the length of the edges decreases by a factor of $2$ at every step.

**Proof:** Consider the shortest path metric $d$ on $G = (V, E)$ with weights given by $c_e$ on edge $e$. The result of Fakcharoenphol et al [17] shows that there exists a distribution on tree metrics such that the average distortion is at most $O(\log n)$. Moreover, each tree in the support of the distribution has depth $\log \Delta$ where $\Delta$ is the diameter of the graph; further, the length of an edge at level $i$ is $2^i$ and the minimum distance between any two vertices is 1. Sample one of the trees from the distribution and solve the online EW group Steiner forest problem on the tree using the algorithm given in Theorem 5.

The competitive ratio of the randomized algorithm, in expectation, is $O(\log^4 n \log k \log \Delta)$. To remove the dependence on $\Delta$ we apply standard doubling tricks. We maintain a guess $\alpha$ of the optimal solution which we double each time we find it to be infeasible. At every update of $\alpha$, we select all edges of the tree of length at most $\alpha/n^2$ in the solution. When a terminal set pair $(S_i, T_i)$ arrives, we delete all edges of the tree of length more than $\alpha$ and the solve the linear program on the forest thus obtained. If in any iteration, we find that the instance is infeasible or if the cost of the linear programming solution is more than $\alpha$, then we double our guess of $\alpha$. This leads a new level of edges to be included in the forest over which we solve the problem. Theorem 3 now follows from standard doubling arguments (omitted due to lack of space).

**Acknowledgement.** The authors would like to thank Moran Feldman for helpful discussions. D. Panigrahi
was partially supported by NSF-STC Award 0939370, and would like to thank David Karger for insightful comments on an earlier draft of this paper.

REFERENCES


