Near-optimal Online Algorithms for Prize-collecting Steiner Problems

MohammadTaghi Hajiaghayi[2] ⋆, Vahid Liaghat[2] ⋆ ⋆⋆, and Debmalya Panigrahi[1] ⋆⋆ ⋆⋆⋆

[1] Computer Science Department, Duke University, Durham, NC 27708
[2] Computer Science Department, University of Maryland, College Park, MD 20742

Abstract. In this paper, we give the first online algorithms with a poly-logarithmic competitive ratio for the node-weighted prize-collecting Steiner tree and Steiner forest problems. The competitive ratios are optimal up to logarithmic factors. In fact, we give a generic technique for reducing online prize-collecting Steiner problems to the fractional version of their non-prize-collecting counterparts losing only a logarithmic factor in the competitive ratio. This reduction is agnostic to the cost model (edge-weighted or node-weighted) of the input graph and applies to a wide class of network design problems including Steiner tree, Steiner forest, group Steiner tree, and group Steiner forest. Consequently, we also give the first online algorithms for the edge-weighted prize-collecting group Steiner tree and group Steiner forest problems with a poly-logarithmic competitive ratio, since corresponding fractional guarantees for the non-prize-collecting variants of these problems were previously known.

For the most fundamental problem in this class, namely the prize-collecting Steiner tree problem, we further improve our results. For the node-weighted prize-collecting Steiner tree problem, we use the generic reduction but improve the best known online Steiner tree result from Naor *et al* [14] on two counts. We improve the competitive ratio by a logarithmic factor to make it optimal (up to constants), and also give a new dual-fitting analysis showing that the competitive ratio holds against the fractional optimum. This result employs a new technique that we call *dual averaging* which we hope will be useful for other dual-fitting analyses as well. For the edge-weighted prize-collecting Steiner tree problem, we match the optimal (up to constants) competitive ratio of $O(\log n)$ that was previously achieved by Qian and Williamson [15] but provide a substantially simpler analysis.

1 Introduction

Over the last two decades, network design problems have been a cornerstone of algorithmic research. The Steiner tree (ST) problem, where the goal is to connect a given set of terminals at minimum cost, and its various generalizations have been central to this research effort. One branch of Steiner problems that has attracted substantial attraction are the so called prize-collecting problems, where the algorithm is permitted to

⋆ Supported in part by NSF CAREER award 1053605, NSF grant CCF-1161626, ONR YIP award N000141110662, DARPA/AFOSR grant FA9550-12-1-0423.
⋆⋆ Part of this work was done while the author was at Microsoft Research, Redmond.
⋆⋆⋆ Supported in part by Duke University startup grant.

violate one or more connection constraints but must pay corresponding *penalties* in the objective function. Prize collecting (PC) Steiner problems were originally motivated by applications in network planning for service providers (see e.g., [11]). They have since become a well-studied branch of approximation algorithms. In this paper, we focus on the classical online model, where the connectivity demands appear over time and must be immediately satisfied. The study of online prize-collecting network design was initiated very recently by Qian and Williamson [15]. This is somewhat surprising on two counts: first, online versions of Steiner problems appear prominently in the algorithmic literature and prize-collecting variants are a natural generalization; and second, the online model is well-motivated in the context of prize-collecting Steiner problems since in practice, new customers appear over time and network providers must upgrade their networks according to the new demands or lose these customers, thereby paying the corresponding penalty in the revenue. In this paper, we provide a simple generic approach to online prize-collecting Steiner problems that reduces these problems to their fractional non-prize-collecting counterparts losing a logarithmic factor in the competitive ratio. Using known results for the online edge-weighted (EW) and node-weighted (NW) Steiner tree problems, this reduction yields algorithms with competitive ratios of $O(\log^2 n)$ and $O(\log^4 n)$ respectively for their prize-collecting variants. Exploring further, we give improved algorithms for both these problems: for the EW problem, we match the competitive ratio of $O(\log n)$ obtained by Qian and Williamson [15], whereas for the NW problem, we obtain a competitive ratio of $O(\log^3 n)$. Both these results are obtained by employing a novel online dual-fitting approach. Our result represents the first algorithm for online NW-PCST that achieves a poly-logarithmic competitive ratio.

The online Steiner tree (ST) problem was originally considered in the EW model, where Imase and Waxman [10] showed that a natural greedy algorithm has a competitive ratio of $O(\log n)$, which is optimal up to constants. This result was generalized to the online EW Steiner forest (SF) problem by Awerbuch *et al* [3], who showed that the greedy algorithm has a competitive ratio of $O(\log^2 n)$. This result was later improved by Berman and Coulston [4] to $O(\log n)$. All the above results can be reproduced using dual-fitting techniques (though the original expositions relied on combinatorial arguments). This immediately shows that the competitive ratios hold against the respective fractional optimums as well. Very recently, Qian and Williamson [15] initiated the study of online PC Steiner problems by providing an $O(\log n)$-competitive algorithm for the online EW-PCST problem. The analysis of this algorithm is quite complicated and uses a dual moat growing approach of Goemans and Williamson [7] and an amortized accounting scheme due to Berman and Coulston [4].

In contrast to EW problems, progress in online NW Steiner problems has been relatively slow. Note that edge weights can be represented by node weights but not vice-versa; so, NW problems are strictly more general. In fact, the NW-ST problem generalizes the set cover problem, for which the first online algorithm with a poly-logarithmic competitive ratio was obtained by Alon *et al* [2]. They introduced an online adaptation of the classical LP relaxation technique, which has since been used extensively in online optimization (see, e.g., the survey by Buchbinder and Naor [5]). In particular, Naor *et al* [14] used this technique in conjunction with structural properties of the NW-ST problem to give an $O(\log^3 n)$-competitive algorithm for the online NW-ST problem. They

left the online NW-SF problem open, which was resolved very recently by Hajiaghayi *et al* [9] who obtained an identical competitive ratio of $O(\log^3 n)$ for the SF problem. However, there is a crucial difference between these two results. Whereas Hajiaghayi *et al* use a dual-fitting approach, which shows that the competitive ratio holds for the fractional optimum as well, Naor *et al* use structural properties of *integral* Steiner trees. Therefore, their results do not hold for the fractional optimum. As described below, this distinction turns out to be important in our work.

Our Techniques and Contributions. Our first contribution is a simple but subtle reduction of online prize-collecting Steiner problems to their respective non-prize-collecting fractional variants losing a factor of $O(\log n)$ in the competitive ratio. This reduction is quite generic and can be applied for more general problems than ST and SF. Indeed, this approach can be applied to any problem which demands $\{0,1\}$-connectivity on a family of cuts. This setting includes the T-join problem and group Steiner tree (GST)/ group Steiner forest (GSF) problems as special cases (see Theorem 3 for a formal description). All these problems are instances of covering problems. In our reduction, we run the algorithm of Buchbinder *et al* [5] for solving covering problems *in parallel* with an algorithm for the non-prize-collecting variant (as a black box). At each online step, we first generate an online competitive fractional solution. Then we use the fractional solution to reveal a modified demand to the non-prize-collecting black box and finally output an integral solution for the prize-collecting problem. This reduction is oblivious of the cost model of the input graph, and hence can be applied to both EW and NW problems, thereby yielding online algorithms for various prize-collecting Steiner connectivity problems with poly-logarithmic competitive ratios. Indeed one main obstacle to solving the online prize-collecting variants of these problems is that the known rounding techniques do not seem to be effective for rounding the standard linear relaxation of the problems. For example for the online NW prize-collecting ST, it is not known whether one can solve the fractional LP for PCST and then round it online. This may have been the reason that the only PC result known before this work, i.e, the Qian-Williamson algorithm [15] for online EW PCST, is quite sophisticated. A summary of results that follow from our reduction are shown in Table 1.

	PC ST	PC SF	PC SF in Planar graphs	PC GSF
EW	$O(\log(n))$[15] Simplified in this paper	$O(\log(n))$[15]	$O(\log(n))$[15]	$O(\log^7(n))$
NW	$O(\log^3(n))$	$O(\log^4(n))$	$O(\log^2(n))$	$O(\log^{11}(n))$ (quasi-polynomial)

Table 1. The competitive ratio for online prize-collecting (PC) Steiner problems. Note that the algorithm for NW Group Steiner Forest runs in quasi-polynomial time.

Next, we focus on the online EW-PCST and NW-PCST problems. For these problems, the generic reduction yields competitive ratios of $O(\log^2 n)$ and $O(\log^4 n)$ respectively. We improve both these competitive ratios by a logarithmic factor. For the EW problem, this matches the competitive ratio of the Qian-Williamson algorithm [15] and is optimal up to constants. Further, our analysis is extremely simple and uses a natural

dual-fitting approach. Due to lack of space, we refer the reader to the full version of the paper for the EW algorithm.

Theorem 1 (also in [15]). *The online edge-weighted prize-collecting Steiner tree problem admits an $O(\log(n))$-competitive algorithm.*

For the online NW-PCST problem, we use the generic reduction, but give an online algorithm for the NW-ST problem that has the optimal competitive ratio of $O(\log^2 n)$ *against the fractional objective*. While it is relatively straightforward to use a re-scaling argument for improving the integral competitive ratio of the algorithm of Naor *et al* [14] by a logarithmic factor, a similar improvement for the (fractionally competitive) algorithm of Hajiaghayi *et al* [9] has fundamental difficulties. So, we first design a novel dual-fitting analysis of the algorithm of Naor *et al*, thereby proving that the competitive ratio of the algorithm now holds against the fractional optimum. However, using this new analysis, we can no longer use the re-scaling argument that improved a logarithmic factor for the integral algorithm. To overcome this difficulty, we introduce a new concept that we call *dual averaging*.

The celebrated moat-growing method of Agrawal, Klein, and Ravi [1] and Goemans and Williamson [7] has been extensively studied for various Steiner connectivity problems. A general pattern in different variants of this method is as follows. We start by growing a moat over every terminal. When two moats collide, this signifies an opportunity for connecting the terminals at the center of the moats thus merging the moats. Since the moats have a dual vector interpretation, by weak duality one can charge the cost of an algorithm to the growth of the moats. When used in a dual-fitting argument, the crux of the analysis is to show that the dual moats do not intersect. In the NW setting, this turns out to be even more difficult since the next terminal that arrives might quickly collide with polynomially many disks at the same vertex thereby making merges of the moats infeasible. We circumvent this problem by introducing a *thinness* factor for the moats. For $\tau \in (0,1]$, a τ-thin dual moat is obtained by scaling the dual variables corresponding to a moat by the factor τ. This allows us to have several *overlapping* moats; this is in contrast to standard dual-fitting methods in which the dual moats are disjoint. The strength of this natural modification is that one can exploit it to show that certain *structural properties* may hold on *average*, although they may not hold for every dual moat independently. For example, consider the feasibility of a dual vector. A reader familiar with the standard dual program for ST may recall that every moat has a load on vertices inside or on the boundary of a moat. It often happens that for every vertex, a few moats have a high load on the vertex while the load of the rest of the moats is negligible. However, for different vertices, the moats with a high load might be different. By considering the proper thinness for the moats, one can balance the loads simultaneously for every vertex to ensure feasibility of the dual moats. We refer the reader to Section 3 for a formal discussion of this approach.

Theorem 2. *The online node-weighted Steiner tree problem admits an $O(\log^2(n))$-competitive algorithm. Moreover, the competitive ratio holds with respect to the optimal fractional solution.*

Note that the above algorithm is optimal in its competitive ratio since there is a known lower bound of $\Omega(\log^2 n)$ [2,13] for the online set cover problem, which is a special case of the online NW-ST (and thus online NW-PCST) problem.

Applying the reduction in Theorem 3 to the algorithm in Theorem 2, we obtain an improved competitive ratio for the online NW-PCST problem. Furthermore, very recently, Hajiaghayi *et al* [9] gave an algorithm with a tight competitive ratio of $O(\log(n))$ for the NW ST problem for planar graphs and more generally graphs excluding a fixed graph as a minor. Their results are based on a primal-dual technique and thus the competitive ratio is w.r.t. the fractional optimum. Therefore we get the following results for the prize-collecting counterparts of these problems.

Corollary 1. *The online NW prize-collecting ST problem admits an $O(\log^3(n))$-competitive algorithm in general graphs. When restricted to graphs excluding a fixed graph as a minor, the problem admits an $O(\log^2(n))$-competitive algorithm.*

In the offline paradigm, algorithms for prize-collecting problems are used at the heart of algorithms for other connectivity problems. An important branch of such problems are *budgeted*[3] and *quota*[4] problems. The key to solving these problems is to design Lagrangian multiplier preserving (LMP) approximation algorithms for PCST. Indeed very recently, Konemann *et al.* [12] gave an LMP $O(\log(n))$-approximation algorithm for offline NW PCST. Therefore, a natural question is whether one can hope for LMP *online* algorithms with poly-logarithmic competitive ratio. We show this is impossible even for the case of online edge-weighted Steiner tree. In the full version of the paper, we show a lower bound of $\Omega(n)$ for the LMP competitive ratio of (randomized) algorithms for online EW-PCST.

2 Online Prize-Collecting Network Design

Let $G = (V,E)$ be an undirected graph. For a set $S \subseteq V$, let $\delta(S) \subseteq V \backslash S$ denote the neighbors of S. Given a $\{0,1\}$-function $f : 2^V \to \{0,1\}$, a *demand system* with respect to f is defined as the following system of inequalities over a set of variables $\mathbf{x}(v)$ for every $v \in V$.

$$\sum_{v \in \delta(S)} \mathbf{x}(v) \geq f(S) \qquad \text{for every } S \subset V, S \neq \phi$$
$$\mathbf{x}(v) \in [0,1]$$

We say a $\{0,1\}$-function f is *efficient* if the following properties hold: (i) $f(\phi) = f(V) = 0$; (ii) for every $S \subseteq V$, $f(S)$ can be computed in polynomial time; and (iii) there exists a separation oracle such that for any vector $\mathbf{x} \in [0,1]^V$, it outputs a set S with $\sum_{v \in \delta(S)} \mathbf{x}(v) < f(S)$ iff there is a violated constraint. The oracle should run in polynomial time w.r.t. $|V|$.

As we will see, the last two properties are required so that our reduction runs in polynomial time. Although the first property is not necessary, it makes the demand

[3] Given an upper limit on the weight, the goal is to maximize the prize of the connected vertices.
[4] We want the minimum-weight subgraph that gathers at least a given amount of prize.

system well defined even if we consider the constraints corresponding to $S = \phi$ and $S = V$ in the system. In the rest of the section, we use $\mathscr{F}$ to refer to a family of efficient functions. Furthermore, we use $\mathscr{G}$ to refer to a family of node-weighted graphs $G = (V,E,w)$ where for $v \in V$, $w_v \in \mathbb{R}_{\geq 0}$.

Problem. A *network design problem* (ND) with respect to $\mathscr{F}$ and $\mathscr{G}$ is defined as follows. Let $G = (V,E,w)$ be a node-weighted graph in $\mathscr{G}$. Given a sequence of functions $f_1,\ldots,f_k \in \mathscr{F}$, the goal is to find a minimum-weight vector $\mathbf{x} \in \{0,1\}^V$ that simultaneously satisfies the demand systems for every function f_i. The ND problem can be formulated as the following integer program (IP). Throughout the paper, for an integer k, let $[k]$ denote $\{1,\ldots,k\}$.

$$\begin{aligned} &\text{minimize} \sum_{v\in V} w_v \mathbf{x}(v) && (\mathbb{ND})\\ \forall S \subseteq V, i \in [k] &\sum_{v\in\delta(S)} \mathbf{x}(v) \geq f_i(S) \\ &\mathbf{x}(v) \in \{0,1\} \end{aligned}$$

Given a feasible solution $\mathbf{x}$, we define $cost(\mathbf{x})$ as the total *weight* of $\mathbf{x}$, i.e, $\sum_{v\in V} w_v \mathbf{x}(v)$.

In a *prize-collecting network design problem* (PCND) w.r.t. $\mathscr{F}$ and $\mathscr{G}$, we are given $G = (V,E,w) \in \mathscr{G}$ and a sequence of *demands* $(f_1,\pi_1),\ldots,(f_k,\pi_k)$ where $f_i \in \mathscr{F}$ and $\pi_i \in \mathbb{R}_{\geq 0}$. For every demand (f_i,π_i), we need to either satisfy the demand system w.r.t. f_i or pay the *penalty* π_i. In other words, we need to find an optimal solution to the following IP.

$$\begin{aligned} &\text{minimize} \sum_{v\in V} w_v \mathbf{x}(v) + \sum_{i\in[k]} \pi_i \mathbf{z}(i) && (\mathbb{PCND})\\ \forall S \subseteq V, i \in [k] &\sum_{v\in\delta(S)} \mathbf{x}(v) \geq f_i(S)(1-\mathbf{z}(i)) \\ &\mathbf{x}(v), \mathbf{z}(i) \in \{0,1\} \end{aligned}$$

Given a feasible solution $(\mathbf{x},\mathbf{z})$ to the IP, we define $cost(\mathbf{x},\mathbf{z})$ as the weight of $\mathbf{x}$ plus the total penalty $\sum_{i\in[k]} \pi_i \mathbf{z}(i)$.

In what follows, we denote the cost of optimal solutions to the programs $\mathbb{ND}$ and $\mathbb{PCND}$ by OPT_{ND} and OPT_{PCND}. One can also relax the integrality constraints in both IPs to get corresponding linear relaxations. We denote the cost of the optimal fractional solutions of the corresponding linear programs (LP) by OPT^*_{ND} and OPT^*_{PCND}.

Online Setting. In the online variants of network design problems and their prize-collecting counterparts, demands arrive sequentially. However, we assume that the node-weighted graph $G = (V,E,w)$ is known in advance. More precisely, in an online prize-collecting network design problem (OPCND), at time $t \in [k]$, a new demand (f_t,π_t) arrives and we need to output a feasible solution $(\mathbf{x}^t,\mathbf{z}^t)$ for the integer program $\mathbb{PCND}$. The decisions are online in the sense that an online algorithm may only increase the values of the variables, i.e., for every $t' < t$, $\mathbf{x}^{t'}(v) \leq \mathbf{x}^t(v)$ and $\mathbf{z}^{t'}(i) \leq \mathbf{z}^{t'}(i)$, for every $v \in V$ and $i \in [k]$.

Consider an algorithm ALG for OPCND and a sequence of demands $\rho = (f_1,\pi_1),\ldots,(f_k,\pi_k)$. Let $\text{ALG}(\rho)$ denote the cost of the output of ALG on the online input ρ, i.e., $\text{ALG}(\rho) = cost(\mathbf{x}_k,\mathbf{z}_k)$. ALG is *$\alpha$-competitive* w.r.t. $\mathscr{G}$ and $\mathscr{F}$,

if for every $G \in \mathscr{G}$ and every sequence of demands $\rho = (f_1, \pi_1), \ldots, (f_k, \pi_k)$ where $f_i \in \mathscr{F}$, we have $\text{ALG}(\rho) \leq \alpha OPT_{PCND}$. ALG is *strongly α-competitive*, if for every ρ, $\text{ALG}(\rho) \leq \alpha OPT^*_{PCND}$. One can define similar notations for online network design (OND) problems by dropping penalties and replacing PCND indices by ND. The main result of this section is the following reduction.

Theorem 3. *Let $\mathscr{G}$ and $\mathscr{F}$ respectively denote a family of graphs and a family of feasible functions. Given a strongly α-competitive algorithm for an online network design problem (OND) w.r.t. $\mathscr{G}$ and $\mathscr{F}$, one can derive a strongly competitive algorithm for the corresponding OPCND with a competitive ratio of $\alpha O(\log(|V|))$.*

Before we prove Theorem 3, we need to recall the following theorem by Buchbinder *et al* [6] (later improved by Gupta and Nagarajan [8][5]). Consider a minimization LP in the form that given a vector c and a matrix A, minimizes $c \cdot \mathbf{x}$ subject to $A\mathbf{x} \geq \mathbf{1}$.[6] A *covering LP* is a special case where all the entries of A are non-negative. In an *online covering problem*, the vector c is known in advance, however, the covering constraints arrive online. After the arrival of a new constraint, the online algorithm needs to output a (fractional) feasible solution without decreasing the previous values of variables.

Theorem 4 (Theorem 4.2 of [6]). *Let n be the number of variables. There exists an algorithm for the online covering problem which finds a fractional solution with the cost within $O(\log(n))$ factor of the optimal fractional solution. Furthermore, the algorithm only increases a variable if it has a positive coefficient in the new constraint.*

Consider a non-trivial constraint[7] of the program $\mathbb{PCND}$ corresponding to a function f_i and a subset S. It can be re-written in the following standard format:

$$\mathbf{z}(i) + \sum_{v \in \delta(S)} \mathbf{x}(v) \geq 1$$

Thus all the constraints are covering constraints. Suppose we want to solve OPCND fractionally. We note that OPCND is not formally a special case of the online covering problem in two aspects. First, the objective function is not fully known, i.e., the prizes are revealed online. Second, in OPCND all constraints corresponding to f_i are revealed at the same time while in an online covering problem, we assume that the constraints are revealed one by one. The former is easy to handle since by Theorem 4, the algorithm of Buchbinder *et al* [5] only changes a variable when it has a positive coefficient in a newly arrived constraint. Thus the variable $\mathbf{z}(i)$ changes only in step i after receiving the demand (f_i, π_i). The second discrepancy can be handled by using the efficiency of function f_i. At any step, let $(\mathbf{x}^*, \mathbf{z}^*)$ denote the current fractional solution. While the solution is not feasible, we find an infeasible constraint and reveal it to the online covering algorithm. This can be done in polynomial time since f_i is efficient. We continue this process until all the constraints are feasible.

[5] The competitive ratio in [8] is improved to $O(\log(k))$ where k is the maximum number of non-zero entries in a row.

[6] Let $\mathbf{1}$ and $\mathbf{0}$ denote the vectors where all the entries are one and zero, respectively.

[7] We say a constraint is trivial if $f_i(S) = 0$

Therefore the algorithm of Buchbinder and Naor can be applied to OPCND to obtain a fractional solution $(\mathbf{x}^*, \mathbf{z}^*)$ such that by Theorem 4, $cost(\mathbf{x}^*, \mathbf{z}^*) \leq OPT^*_{PCND} O(\log(|V|))$. Note that although the algorithm may increase the value of a variable beyond one, this can be ignored since feasibility is maintained even if we decrease such variables to one.

Algorithm. Let ALG_{OND} be a strongly α-competitive algorithm for OND. The following algorithm for OPCND realizes Theorem 3. Let $\rho = (f_1, \pi_1), \ldots, (f_k, \pi_k)$ denote the online input. We run the online fractional algorithm of Buchbinder *et al* and an instance of ALG_{OND} in parallel. At any time step, let $(\mathbf{x}^*, \mathbf{z}^*)$ denote the (partial) output of the fractional algorithm and let $\mathbf{x}$ denote the (partial) output of ALG_{OND}, respectively. We also maintain an integral vector $\mathbf{z}$ which shows the integral decisions of our algorithm for paying the penalties of demands that have arrived.

At step i, we receive the new demand (f_i, π_i). We reveal the new demand to the online fraction algorithm which in return updates the values of $(\mathbf{x}^*, \mathbf{z}^*)$. In particular, it sets the first and final value of $\mathbf{z}^*(i)$. Now if $\mathbf{z}^*(i) \geq 1/2$, we pay the penalty of the new demand and set $\mathbf{z}(i) = 1$. Otherwise, we set $\mathbf{z}(i) = 0$, and we reveal the function f_i to the instance of ALG_{OND}. At the end of iteration, we report $(\mathbf{x}, \mathbf{z})$ as the output of our algorithm. For a detailed analysis, the reader is referred to the full version of the paper.

3 An Asymptotically Optimal Algorithm for Online NW ST

The online node-weighted Steiner tree (NW-ST) problem is a fundamental OND problem: given a vertex root, every input function f_i characterizes the cuts that separate a vertex t_i from the root.

The Online Node-weighted Steiner Tree problem. We are given an undirected connected graph $G = (V, E)$ where w_v is the weight of vertex $v \in V$. Let $n = |V|$. The online input comprises a sequence of vertices $t_0, t_1, t_2, \ldots, t_k$ where $t_i \in V$. The output comprises a sequence $H_0, H_1, H_2, \ldots, H_k$, where (i) H_i is a connected subgraph of G; (ii) the terminals $\{t_0, t_1, t_2, \ldots, t_i\}$ are connected in H_i; and (iii) H_i is a subgraph of H_{i+1}. The objective is to minimize the total weight of vertices in H_k. Without loss of generality, we assume that the weight of every terminal is zero[8]. For simplicity, we will assume that the cost of the optimal solution is n and for every vertex v, $w_v \in (0, n]$. This is wlog up to a constant factor loss in the competitive ratio[9].

Our algorithm follows the approach of Naor *et al* for solving the problem via an instance of a facility location problem. Although our algorithm is very similar to that of Naor *et al*, our analysis is quite different; while they use a combinatorial fact to prove the competitive ratio of the algorithm, we use the technique of dual averaging that we alluded to in the introduction. This allows us to establish the tight competitive ratio of $O(\log^2(n))$ with respect to the fractional solution.

[8] For every vertex v, attach a dummy vertex ρ_v with weight zero to v. Upon receiving a terminal t, we virtually assume that the terminal is ρ_t.

[9] By an online doubling strategy, we may assume that we know the objective value α of an optimal solution. We multiply all vertex weights by n/α so that the cost of the optimal solution is n. All vertices v with $w_v > n$ are discarded, while we set $x_v = 1$ for those satisfying $w_v \leq 1$.

An Auxiliary Linear Program. Let Γ denote the set of compound indices $V \cup (V \times [k])$, i.e., for every $v \in V$ and $i \in [k]$, both $v \in \Gamma$ and $(v,i) \in \Gamma$. A set $S \subseteq \Gamma$ is an *auxiliary cut* for a terminal t_i, if for every vertex v, S contains exactly one of v and (v,i). Let $\mathbf{S}_i$ denote the collection of all auxiliary cuts for t_i, i.e., for $i \in [k]$, $\mathbf{S}_i = \{S \subseteq \Gamma | \forall_{v \in V} \ |S \cap \{v,(v,i)\}| = 1\}$. For every $v \in V$ and $i \in [k]$, let $P_{(v,i)}$ denote a minimum-weight path between t_i and any previous terminal t_j (i.e. $0 \leq j < i$) which goes through v. For every (compound) index $\gamma \in \Gamma$ we define a weight $\mathbf{w}_\gamma$ as follows. For every $v \in V$, let $\mathbf{w}_v = w_v$. For every $v \in V$ and $i \in [k]$, $\mathbf{w}_{(v,i)}$ is the weight of $P_{(v,i)}$ minus the weight of v.

Consider the *auxiliary linear program* ALP (given below) with a variable $\mathbf{x}_\gamma$ for every index $\gamma \in \Gamma$. Let $\mathbf{x}$ be a feasible solution to the Program ALP. Observe that for every $v \in V$ and $i \in [k]$, we may assume $\mathbf{x}_{(v,i)} \leq \mathbf{x}_v$; otherwise by reducing $\mathbf{x}_{(v,i)}$ to $\mathbf{x}_v$ we can decrease the objective value while keeping the solution feasible[10]. Therefore in the rest of section, wlog, we assume that for every feasible solution, $\mathbf{x}_{(v,i)} \leq \mathbf{x}_v$.

$$\text{minimize} \quad \sum_{\gamma \in \Gamma} \mathbf{w}_\gamma \mathbf{x}_\gamma \qquad \text{(ALP)}$$

$$\forall\, i \in [k], S \in \mathbf{S}_i \quad \sum_{\gamma \in S} \mathbf{x}_\gamma \geq 1 \qquad \text{(P1)}$$

$$\mathbf{x}_\gamma \in [0,1]$$

$$\text{minimize} \quad \sum_{\gamma \in \Gamma} \tilde{\mathbf{w}}_\gamma \mathbf{x}_\gamma \qquad \text{(SALP)}$$

$$\forall\, i \in [k], S \in \mathbf{S}_i \quad \sum_{\gamma \in S} \mathbf{x}_\gamma \geq 1 \qquad \text{(P2)}$$

$$\mathbf{x}_\gamma \in [0,1]$$

It is easy to verify that for every *integral* feasible solution $\mathbf{x}$ for Program ALP, there exists an integral solution for the Steiner tree instance having cost at most the same as the objective value of the program.

A main obstacle in solving the online ST problem is that the known rounding methods are not effective in rounding a fractional solution of the linear relaxation of standard programs for ST. Indeed an important property of the auxiliary LP is that a standard rounding method similar to the Set Cover problem can be used to round the solution by losing (roughly) a logarithmic factor. However, although one can obtain an integral solution for ST with the same cost as that for Program ALP, the converse does not hold. Naor *et al* [14] use a combinatorial decomposition of an *integral* solution for ST to show that the converse holds if one is willing to incur a factor of $O(\log^2(n))$ in the cost. This combinatorial fact does not have a fractional counterpart which is crucial to our reduction in solving PCST. Furthermore, using Program ALP leads to a competitive ratio of $O(\log^3(n))$ after applying the rounding method, which is off by a logarithmic factor from the known lower bound. We overcome both obstacles by using a dual averaging argument to show a similar relationship between *fractional* solution for ST and that for a *scaled auxiliary LP*.

We define a *scaled weight* $\tilde{\mathbf{w}}$ over the set of compound indices Γ as follows. For every $v \in V$ and $i \in [k]$, let $\tilde{\mathbf{w}}_{(v,i)} = \mathbf{w}_{(v,i)}$, while for every $v \in V$, let $\tilde{\mathbf{w}}_v = w_v \log(n)$.[11] The scaled auxiliary program SALP is given above. We split the objective function of this LP into two parts. For a feasible vector $\mathbf{x}$ for SALP, let the *facility cost*, $\mathit{FacCost}(\mathbf{x}) =$

[10] Recall that for every auxiliary cut S that contains (v,i), there exists a cut S' that replaces (v,i) with v. Thus if constraint P1 is feasible for S', by reducing $\mathbf{x}_{(v,i)}$ to $\mathbf{x}_v$, the constraint corresponding to S remains feasible.

[11] In the rest of the section, the base of all logarithmic terms is 2.

$\sum_{v\in V} \tilde{\mathbf{w}}_v \mathbf{x}_v$ and let the *connection cost*, $ConCost(\mathbf{x}) = \sum_{v\in V, i\in[k]} \tilde{\mathbf{w}}_{(v,i)} \mathbf{x}_{(v,i)}$. We may drop $\mathbf{x}$ from the notation when the vector is clear from the context. Observe that a feasible solution for SALP yields a feasible solution for NW ST with total cost at most $\frac{FacCost}{\log(n)} + ConCost$.

Algorithm. We first find an online[12] fractional solution for SALP using the method of multiplicative updates (a formal discussion of this method is presented in the full version of the paper). We then show the objective value of this fractional solution is within $O(\log^2(n))$ factor of the optimal fractional solution for ST. Note that the objective function of the program uses the scaled weights for vertices. A feasible solution for the *scaled program* can be rounded online with an additional loss of a *constant* factor using the standard rounding techniques. We will not describe this rounding procedure here; we refer the reader to Section 2 of [14].

Analysis. For a subset of vertices $S \subset V$, let $\delta(S) \subseteq V \backslash S$ denote the neighbors of S. Let $\mathscr{S}$ denote the collection of subsets of vertices that separate a subset of terminals from the terminal t_0, i.e, $S \in \mathscr{S}$ if and only if $S \cap \{t_1, \ldots t_k\} \neq \phi$ and $t_0 \notin S$. Consider the natural LP relaxation for the NW ST problem in which there is a flow of one going out of every set in $\mathscr{S}$. The primal dual pair for this LP is as follows.

$$\begin{array}{llll} \textit{minimize} & \sum_{v\in V} w_v x_v & \textit{maximize} & \sum_{S\in\mathscr{S}} y(S) \\ \forall S \in \mathscr{S} & \sum_{v\in\delta(S)} x_v \geq 1 & \forall v & \sum_{S\in\mathscr{S}: v\in\delta(S)} y(S) \leq w_v \\ & x_v \geq 0 & & y(S) \geq 0 \end{array} \tag{D1}$$

For two vertices u and v, let $d(u,v)$ denote the weight of shortest path between u and v *excluding* the weight of the endpoints. We borrow the notation of Hajiaghayi *et al* [9] for disks. A *painting* is a function $p : V \to \mathbb{R}_{\geq 0}$. A painting is *valid* if $p(v) \leq w_v$ for every vertex v. Intuitively, every vertex has an area equal to its weight and a painting is a (partial) coloring of these areas. The union of a set of paintings $p_1, \ldots, p_\ell$ is the painting p with $p(v) = \sum_{i\in[\ell]} p_i(v)$ for every $v \in V$.

Recall that the weight of a terminal is zero. A *disk* of radius r centered at terminal t, is a painting in which the area within a radius r of t is colored, i.e.,

$$p(v) = \begin{cases} w_v & \text{if } d(t,v) + w_v \leq r \\ r - d(t,v) & \text{if } d(t,v) + w_v > r \text{ but } d(t,v) \leq r \\ 0 & \text{if } d(t,v) \geq r \end{cases}$$

A disk p centered at a terminal t_i for $i \in [k]$ is *feasible* if $p(t_0) = 0$.

A *disk vector* corresponding to a disk p of radius r centered at a terminal t is a dual vector y generated by the following deterministic process:

1: Initialize $y = \mathbf{0}$ and $U = \{t\}$.
2: **while** the total dual objective $\sum_{S\in\mathscr{S}} y(S)$ is less than r **do**

[12] In the online setting, at time step $i \in [k]$, the constraints for sets $S \in \mathbf{S}_i$ are revealed and the algorithm should output a feasible solution. However, the online algorithm may only increase the previous values of variables in each time step.

3: Continuously increase $y(U)$ until $\sum_{S\in\mathscr{S}} y(S)$ reaches r, or for a vertex v the dual constraint $D1$ becomes tight.
4: If the latter happens, add v to U.

For a dual vector y, define the *load* of y on a vertex v as $\sum_{S\in\mathscr{S}:v\in\delta(S)} y(S)$. The construction of a disk vector directly yields the following lemma (due to lack of space, we refer the reader to the full version of the paper for the proofs in this section).

Lemma 1. *Let y be a disk vector corresponding to a disk p. For every $v \in V$, the load of y on v is exactly $p(v)$.*

Corollary 2. *Let y be a disk vector corresponding to a disk of radius r centered at t. For every vertex $v \in V$ on which the load of y is strictly positive, we have $r \geq d(t,v)$. Furthermore, y is a feasible dual vector if and only if the disk is feasible, i.e., $r \leq d(t,t_0)$.*

For an arbitrary *thinness* factor $\tau \in (0,1]$, a τ-thin disk vector is a dual vector y' obtained by scaling the disk vector y by a factor of τ, i.e., $y'(S) = \tau y(S)$ for every $S \in \mathscr{S}$. When the thinness factor is clear from the context, we may refer to y' as simply a thin disk vector. We note that a τ-thin disk vector is feasible if and only if the corresponding (1-thin) disk vector is feasible. Observe that the total dual objective value of a τ-thin disk vector with radius r is exactly $r \times \tau$. Similar to paintings, the union of a set of disk-vectors $y_1, \ldots, y_\ell$ is the dual vector y where $y(S) = \sum_{i\in[\ell]} y_i(S)$ for every $S \subseteq \mathscr{S}$.

Multiplicative Updates Method. Consider the cost of the fractional solution generated by the multiplicative steps. In what follows, let *opt* denote the cost of the optimal fractional solution for the NW-ST problem. Recall that we assume $opt = n$. In our algorithm, for every $\gamma \in \Gamma$, we initialize $\mathbf{x}_\gamma$ to $\frac{1}{n^3}$ (or to one if $\tilde{\mathbf{w}}_\gamma = 0$). Furthermore, for $v \in V$ and $i \in [k]$, $\tilde{\mathbf{w}}_{(v,i)} \leq n^2$ since we have assumed that the weight of a single vertex is at most n and a simple path may contain at most n vertices. SALP has at most $n^2 + n$ variables. Thus the objective cost of the initialization is at most $(n^2+n)(n^2)\frac{1}{n^3} \leq 2n = 2opt$. Hence the cost of initialization adds a constant factor to the competitive ratio; we will ignore this factor in the remaining analysis.

Using the notion of thin disks, we will now establish the competitive ratio of the algorithm. Consider the cost of the fractional solution generated by the multiplicative steps. Using standard techniques, one can show that to bound the cost of the algorithm, it is sufficient to bound the number of multiplicative steps. Therefore we use a union of a set of disks to account for the *number* of the multiplicative updates.

Lemma 2. *Let FacCost and ConCost respectively denote the facility cost and the connection cost of the output of the algorithm. Then $FacCost + ConCost \leq O(\log^2(n))opt$.*

For $i \in [k]$, let s_i denote the number of multiplicative steps done for terminal t_i. We assume wlog that $s_i \geq 1$. Let $\tau = \frac{1}{4\log(n)}$. For every $i \in [k]$, consider a τ-thin disk vector y_i corresponding to a disk of radius $r_i = \frac{s_i}{5\log(n^3)}$ centered at terminal t_i. We first show for every $i \in [k]$, the thin disk vector y_i is indeed feasible. Note that for terminal t_i and in every auxiliary cut we should have either t_0 or (t_0, i). However, the cost of a terminal, in particular that of t_0, is zero. Hence, $\mathbf{x}_{t_0}$ is initialized to one and thus only (t_0, i) can participate in a multiplicative step. Thus using standard arguments, one can show that

$s_i \leq \tilde{\mathbf{w}}_{(t_0,i)} \log(n^3)$. Recall that $\tilde{\mathbf{w}}_{(t_0,i)}$ is the weight of shortest path between t_i and a terminal t_j for $j < i$, which passes through t_0. Hence by choosing $j = 0$, $\tilde{\mathbf{w}}_{(t_0,i)} \leq d(t_i, t_0)$. Thus the radius of the disk is at most $r_i = \frac{s_i}{5\log(n^3)} \leq \frac{d(t_0,t_i)}{5}$, which by Corollary 2 verifies that y_i is feasible.

Let y denote the union of y_i's for every $i \in [k]$. We prove Lemma 2 by showing that the dual vector y is feasible. Finally as mentioned before, by using a standard rounding method, Theorem 2 follows. We refer the reader to the full version of the paper for a formal analysis.

Acknowledgement

D. Panigrahi would like to thank Niv Buchbinder for suggesting the re-scaling trick to remove an additional logarithmic factor from the competitive ratio of the integral algorithm of Naor *et al* [14].

References

1. Ajit Agrawal, Philip Klein, and R. Ravi. When trees collide: an approximation algorithm for the generalized steiner problem on networks. In *STOC*, pages 134–144, 1991.
2. Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. The online set cover problem. *SIAM J. Comput.*, 39(2):361–370, 2009.
3. Baruch Awerbuch, Yossi Azar, and Yair Bartal. On-line generalized steiner problem. *Theor. Comput. Sci.*, 324(2-3):313–324, 2004.
4. Piotr Berman and Chris Coulston. On-line algorithms for steiner tree problems. In *STOC*, pages 344–353, 1997.
5. Niv Buchbinder and Joseph Naor. The design of competitive online algorithms via a primal-dual approach. In *FOCS*, pages 93–263, 2009.
6. Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing problems. *Math. Oper. Res.*, 34(2):270–286, 2009.
7. Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.
8. Anupam Gupta and Viswanath Nagarajan. Approximating sparse covering integer programs online. In *ICALP*, pages 436–448, 2012.
9. MohammadTaghi Hajiaghayi, Vahid Liaghat, and Debmalya Panigrahi. Online node-weighted steiner forest and extensions via disk paintings. In *FOCS*, pages 558–567, 2013.
10. Makoto Imase and Bernard M. Waxman. Dynamic steiner tree problem. *SIAM J. Discrete Math.*, 4(3):369–384, 1991.
11. David S. Johnson, Maria Minkoff, and Steven Phillips. The prize collecting steiner tree problem: theory and practice. In *SODA*, pages 760–769, 2000.
12. Jochen Könemann, Sina Sadeghian Sadeghabad, and Laura Sanità. An LMP $O(\log n)$-approximation algorithm for node weighted prize collecting steiner tree. In *FOCS*, pages 568–577, 2013.
13. Simon Korman. On the use of randomization in the online set cover problem. *M.S. thesis, Weizmann Institute of Science*, 2005.
14. Joseph Naor, Debmalya Panigrahi, and Mohit Singh. Online node-weighted steiner tree and related problems. In *FOCS*, pages 210–219, 2011.
15. Jiawei Qian and David P. Williamson. An $O(\log n)$-competitive algorithm for online constrained forest problems. In *ICALP*, pages 37–48, 2011.