Online Node-weighted Steiner Forest and Extensions via Disk Paintings

MohammadTaghi Hajiaghayi
Computer Science Dept.
Univ of Maryland
College Park, MD, USA
Email: hajiagha@cs.umd.edu

Vahid Liaghat
Computer Science Dept.
Univ of Maryland
College Park, MD, USA
Email: vliaghat@cs.umd.edu

Debmalya Panigrahi*
Computer Science Dept.
Duke University
Durham, NC, USA
Email: debmalya@cs.duke.edu

***Abstract*—We give the first polynomial-time online algorithm for the node-weighted Steiner forest problem with a poly-logarithmic competitive ratio. The competitive ratio of our algorithm is optimal up to a logarithmic factor. For the special case of graphs with an excluded fixed minor (e.g., planar graphs), we obtain a logarithmic competitive ratio, which is optimal up to a constant, using a different online algorithm. Both these results are obtained as special cases of generic results for a large class of problems that can be encoded as online $\{0,1\}$*-proper functions*.**

Our results are obtained by using a new framework for online network design problems that we call *disk paintings*. The central idea in this technique is to amortize the cost of primal updates to a set of carefully selected mutually disjoint fixed-radius dual *disks* centered at a subset of terminals. We hope that this framework will be useful for other online network design problems.

I. Introduction

Steiner problems, where the goal is to find the minimum weight subgraph of a given (undirected) graph that satisfies a given set of connectivity requirements, form a fundamental class of optimization problems that have attracted substantial attention over the last few decades. The *Steiner tree* (ST) problem—which asks for the minimum weight subgraph connecting a given set of vertices called *terminals*—is perhaps the most representative problem in this class. This paper deals with its well-studied generalization called the the *Steiner forest* (SF) problem where the connectivity requirement is represented by a set of vertex pairs called *terminal pairs* that need to be individually connected in the classical *online* model, i.e., the input graph is given offline but the terminal pairs arrive sequentially in online steps. The selected subgraph starts off as the empty subgraph, but has to be augmented to satisfy the new connectivity constraint in each online step. We measure the performance of our algorithms using the classical notion of *competitive ratio*, i.e., the maximum ratio (over all input sequences) of the weight[1] of the algorithmic solution to that of the offline optimum.

Steiner problems have typically been studied in two weight models: the *edge-weighted* (EW) model and the *node-weighted* (NW) model, depending on whether the weight function is defined on the edges or the vertices respectively. The NW model is more general since an edge of weight w in the EW model can be replaced in the NW model by two edges connected by a node of weight w; therefore, algorithmic results in the NW model also apply to the EW model.

The online ST problem was originally considered in the EW model, where Imase and Waxman [1] showed that the greedy strategy of adding the minimum cost subset of edges in each online step obtains a competitive ratio of $O(\log k)$, which is optimal up to constants. (Throughout this paper, m and n will represent the number of edges and vertices in the input graph respectively, while k will represent the number of terminals.) This result was generalized to online EW SF by Awerbuch *et al* [2], who showed a competitive ratio of $O(\log^2 n)$ for the greedy algorithm. This result was later improved by Berman and Coulston [3] who proposed an online algorithm based on the primal dual framework [4], [5] that achieves a competitive ratio of $O(\log n)$.

However, much less progress has been reported for NW versions of these problems. Unlike in the EW model, the NW ST problem generalizes the set cover problem. The first algorithm with a poly-logarithmic competitive ratio for the online set cover problem was proposed by Alon *et al* [6] who introduced an online adaptation of the classical LP relaxation technique to solve this problem. Recently, Naor *et al* [7] used this technique in conjunction with structural properties of the NW ST problem to give an $O(\log n \log^2 k)$-

*Part of this work was done while the author was at Microsoft Research, Redmond.

[1]expected weight, if the algorithm is randomized

competitive algorithm for the online NW ST problem. They also presented a poly-logarithmic competitive algorithm for the online NW SF problem, but the running time of the algorithm is quasi-polynomial. It is important to note that there is a qualitative difference between quasi-polynomial time algorithms and polynomial time algorithms for NW graphs. NW undirected graphs can be reduced to EW *directed* graphs, which allows for the application of known techniques for obtaining a poly-logarithmic approximation for the directed Steiner tree problem (see, e.g., Charikar *et al* [8]) in quasi-polynomial time. In fact, the algorithm of Naor *et al* for the online NW SF problem implicitly uses this transformation. In contrast, obtaining a poly-logarithmic approximation for the directed Steiner tree problem in polynomial time is a major open question. Therefore, polynomial-time algorithms for NW Steiner problems must develop novel techniques that successfully bypass this reduction to directed graphs. Naor *et al* developed such a set of techniques for the online NW ST problem, and posed the more general NW SF problem as their main open question. We resolve this question in this paper by introducing a new generic technique that we call *disk paintings* and using it in conjunction with a connection between the NW ST problem and the non-metric facility location problem previously observed by Naor *et al*. The competitive ratio of our algorithm is $O(\log n \log^2 k)$, which matches the competitive ratio for the NW ST problem [7] and is optimal up to a logarithmic factor due to a lower bound of $\Omega(\log n \log k)$ [6], [7]. An interesting observation is that our algorithm yields a distinct algorithm for the online NW ST problem from that of Naor *et al* when applied to NW ST instances.

We show that in addition to solving the online NW SF problem on arbitrary graphs, disk paintings can be used to exploit combinatorial properties of graphs with an excluded constant-sized minor (such as *planar graphs* and more generally, graphs that can be embedded in surfaces of bounded genus) in online Steiner problems. We obtain an $O(\log n)$ competitive algorithm (optimal up to constants) for the online NW SF problem, which improves upon the online NW SF algorithm for general graphs described above. To the best of our knowledge, this is the first instance of an online network design problem where planarity (or more generally, exclusion of a fixed minor) has been successfully exploited to obtain an improved competitive ratio compared to the best known (in this case, also the best achievable) result for arbitrary graphs. We hope that disk paintings will be useful for other online network design problems in the future.

We also extend our results (both for general graphs and for graphs with an excluded fixed minor) to all $\{0,1\}$-*proper functions* [5], which includes the SF problem as a special case but also includes other problems such as T-join, point-to-point connection problems, lower capacitated partitioning, and location-design/location-routing problems. We will formally describe a few examples of such problems later; for a detailed description of problems that can be represented by proper functions, the reader is referred to [5].

Before describing our results and techniques in detail, we remark that offline NW Steiner problems have been studied previously. Klein and Ravi [9] introduced the notion of spider decompositions to give an $O(\log k)$-approximation for the NW ST problem, which is optimal up to constants. (The constant in the approximation ratio was improved later [10].) The Klein-Ravi result also extends to the SF problem, and more generally, to all proper functions. Recently, Demaine *et al* [11] gave an $O(1)$-competitive algorithm for the NW SF problem in planar graphs, where the constant was further improved by Moldenhauer [12] and the algorithm generalized to higher connectivity by Chekuri *et al* [13]. In fact, our online algorithm for planar graphs employs the disk painting technique in conjunction with structural ideas developed in [11] for the corresponding offline problem. Several other offline NW network design problems have been considered in the literature (see, e.g., [14], [15], [16]).

A. Problems

We are now ready to formally define the SF problem (from now on, the problems we refer to are online and NW, unless otherwise stated). Let $G = (V,E)$ be an undirected graph containing n vertices ($|V| = n$) and m edges ($|E| = m$), and let $w : V \to \mathbb{R}^{\geq 0}$ be the node weight function. The subgraph induced on G by a subset of vertices S is denoted by $G[S]$, and has weight $w(S) = \sum_{v \in S} w(v)$. The graph G and the weight function w are given offline.

In the SF problem, we are given a new pair of terminals (also called a *demand*) $s_h, t_h \in V$ (s_h and t_h are called the endpoints of the demand) in online step h. The algorithm maintains a subset of vertices X such that $G[X]$ connects all pairs of terminals that have arrived thus far. In response to the arrival of a new terminal pair, the vertex set X can be augmented to ensure this property. The goal is to minimize the weight of X, i.e., $w(X)$. We note that the ST problem is a special case of the SF problem where one terminal in each terminal pair is a fixed vertex.

We also consider the more general class of Steiner problems that can be represented by $\{0,1\}$-proper functions [5]. We adapt an approach due to Goemans and Williamson [5] and Demaine, Hajiaghayi, and Klein [11] to define a general family of node-weighted network-design problems. Let $G = (V,E)$ be a connected graph with a node-weight function w. Following Demaine et al. [11], a $\{0,1\}$-function $f : 2^V \to \{0,1\}$ is *proper* for node-weighted problems if the following properties hold:

1) **Symmetry** $f(S) = f(V \setminus S)$ for every $S \subseteq V$.
2) **Disjointness** If S_1 and S_2 are disjoint, then $f(S_1) = f(S_2) = 0$ implies $f(S_1 \cup S_2) = 0$.
3) **Nullity** $f(\phi) = f(V) = 0$.
4) **Terminality** Every vertex v with $f(\{v\}) = 1$ has weight $w(v) = 0$.
5) **Efficiency** For any subset S, the value $f(S)$ is computable in polynomial time.

A proper function defines a family of cuts on the graph and the *network design* problem asks for a minimum-cost subgraph X which covers all the cuts, i.e., $G[X]$ must contain an edge of the cut $(S, V \setminus S)$ for every set S with $f(S) = 1$. For a subset of vertices $X \subseteq V$, we may use the set X and the subgraph $G[X]$ interchangeably, when it is clear from the context. A set X *satisfies* a proper function f, if X covers all the cuts defined by f. Equivalently, we say X is *feasible* if it satisfies f.

Following [17], we extend the notion of proper functions to the online setting where in each online step h, a proper function q_h on the vertices is presented. Let f_h denote the *cumulative function* of the step, i.e., $f_h(S) = \max_{j \le h} q_j(S)$ for every $S \subseteq V$. The following shows that a cumulative function is proper too (proof in the full version).

Proposition 1. *Let q_1 and q_2 be proper functions for a set of vertices V with node weights w. Let $f(S) := \max\{q_1(S), q_2(S)\}$ for every $S \subseteq V$. The function f is proper.*

In the online variant of the problem, the solution produced at the end of online step h must satisfy f_h.

Note that the SF problem can be represented by the following proper function: in online step h with terminal pair (s_h, t_h), for any subset $S \subseteq V$,

$$q_h(S) = \begin{cases} 1, & \text{if } |\{s_h, t_h\} \cap S| = 1 \\ 0, & \text{otherwise.} \end{cases}$$

These functions q_h satisfy properties 1-4 of proper functions by definition. For terminality, all terminals must have zero cost. We show later that this property can be ensured w.l.o.g.

As examples of other problems that can be represented by proper functions, let us now define the point-to-point connection problem [18] and the T-join problem [19]. In the (offline) point-to-point connection problem, we are given a set of terminal pairs (s_h, t_h) (as in SF) but instead of pairing them, they are represented as a set of *sources* X and *sinks* Y such that $|X| = |Y|$. The goal is to find a minimum cost subgraph such that the number of sources in any connected component equals the number of sinks in the component. In the online version of the problem, every online step comprises a set of sources X_h and a set of sink Y_h such that $|X_h| = |Y_h|$. The corresponding proper function is given by: for any subset $S \subseteq V$,

$$q_h(S) = \begin{cases} 1, & \text{if } |X_h \cap S| \neq |Y_h \cap S| \\ 0, & \text{otherwise.} \end{cases}$$

Note that if $|X_h| = |Y_h| = 1$, then the problem is identical to online SF.

In the (offline) T-join problem, we are given an even set of terminals T and the goal is to find a minimum cost subgraph of the input graph that contains at least one edge for every cut that has an odd number of terminals on both sides. In the online version, each online step adds an even set of vertices T_h to T. The corresponding proper function is given by: for any subset $S \subseteq V$,

$$q_h(S) = \begin{cases} 1, & \text{if } |T_h \cap S| \text{ is odd} \\ 0, & \text{otherwise.} \end{cases}$$

Finally, we need to define H-minor-free graphs. A graph H is a *minor* of graph G if H can be realized from G by the following set of operations: contracting an edge, deleting an edge, or deleting a vertex. As mentioned above, some of the results in this paper will apply to classes of input graphs where a fixed graph H of constant size is not a minor of any graph in the class.

Assumptions. In the rest of paper, w.l.o.g., we assume the terminals are distinct and have weight 0. This can be ensured in the Steiner forest problem by attaching a proxy vertex of weight 0 to every vertex of the graph. In every online step, we interpret the corresponding proxy vertices as the terminal pair. Let T denote the set of vertices with weight 0, i.e., the possible terminals and let k denote the number of terminals that have arrived.

B. Our Results

We show the following result for proper functions.

Theorem 1. *There is a randomized online algorithm with competitive ratio $O(\log^2 k \log n)$ for network design problems characterized by proper functions.*

When applied to the SF problem, we obtain the following corollary.

Corollary 1. *There is a randomized polynomial-time algorithm for the online node-weighted Steiner forest problem that has a competitive ratio of $O(\log n \log^2 k)$.*

This result:

- improves upon the result of Naor *et al* [7] for SF in two ways: their running time was quasi-polynomial and competitive ratio was $O(\log^3 n \log^7 k)$.
- matches the competitive ratio of Naor *et al* for ST up to constants.
- is optimal up to $O(\log n)$ since online set cover (which SF generalizes) has a randomized lower bound of $\Omega(\log n \log k)$ [6], [20].

Next, we obtain the following result for graphs with an excluded fixed minor.

Theorem 2. *There is a deterministic polynomial-time algorithm with a competitive ratio of $O(\log k)$ for H-minor-free node-weighted input graphs, where H is a fixed graph of constant size, for network design problems characterized by proper functions.*

When applied to the SF problem, we obtain the following corollary.

Theorem 3. *There is a deterministic polynomial-time algorithm for the online node-weighted Steiner forest problem that has a competitive ratio of $O(\log k)$ for H-minor-free input graphs, where H is a fixed graph of constant size.*

For H-minor-free graphs, this result:

- improves upon Theorem 1 for SF and upon [7] for ST. (We note the lower bound of $O(\log n \log k)$ does not apply in the H-minor-free case.)
- is optimal up to a constant since there is an $\Omega(\log k)$ lower bound for the online EW ST problem. (This lower bound can be demonstrated by using a *diamond* graph, which is planar. We omit the details for brevity.)

C. Our Techniques: Disk Paintings

The principle of weak duality in minimization problems asserts that the optimal solution to a primal linear program (LP) is lower bounded by *any* feasible solution to the dual LP. This has inspired the classical (offline) primal-dual method [4], [5] where a progressively constructed dual solution guides the choices made by the algorithm in the primal solution. For Steiner problems, the dual solution is constructed by growing *moats* around every terminal. When two moats collide, they are merged by buying the path connecting the corresponding terminals and the merged moat continues growing. Thus, the sets with positive dual variables form a *laminar* family. The crux of the analysis is to charge the cost of purchased paths to the sum of the radii of the moats, and highly relies on the fact that when two growing moats collide, they have roughly the same radii.

However, in online settings, this property cannot be ensured since the terminals are identified sequentially in online steps. In fact, in any online step, there is only one moat that is growing, namely the one containing the new terminal. In spite of this difficulty, the primal-dual framework has recently been utilized for online algorithms in two distinct lines of work: either a dual solution is used to guide the construction of a *fractional* solution to the primal LP, which is then been rounded online to produce an integer solution (see, e.g., the survey by Buchbinder and Naor [21]) or the algorithm maintains a multi-layered collection of laminar families of moats (see, e.g., [17]).

While our technique of disk paintings also utilizes the broad framework of using a dual solution to guide the algorithmic choices, we deviate significantly from these previous approaches in the structure of the dual solution that we construct, which we describe below. A *disk painting* is simply a set of disjoint disks centered at terminals. Since we have NW graphs, disks intersect at vertices rather than edges. In fact, unlike in the EW case, more than two disks can cumulatively cause an intersection at a vertex. To visualize these disks, let us assign a distinct color to every disk and an area equal to its weight to every vertex. Then, a disk may color a vertex either wholly or partially. For example if x, y, z are three vertices of weight $2w$ each connected by edges (x, y) and (y, z), and we add a disk of radius $5w$ centered at x, then the disk colors y fully (i.e. its entire weight) but z only partially (half its weight). An intersection is caused at a vertex when the sum of weights colored by disks containing the vertex (either fully or partially) exceeds the total weight of the vertex.

To formally define disk paintings, we first need to define distances between vertices. For any pair of vertices u, v, let $d^w(u, v)$ denote the weight of the shortest path between u and v (including u, v) w.r.t. a node weight function w. For a set S of vertices, let $d^w(S, v) = \min_{u \in S} d^w(u, v)$. A *painting* is a function $p : V \times \Theta \to \mathbb{R}_{\geq 0}$, where Θ is a set of colors. Let $p(v)$ denote the total colored area of vertex v, i.e., $p(v) = \sum_{\theta \in \Theta} p(v, \theta)$. A painting p is *feasible* if the colored area of a vertex does not exceed its weight, i.e., $p(v) \leq w(v)$ for every vertex v. The *union* of a set

of paintings $p_1, p_2, \ldots$ is the painting $p = \sum_i p_i$ (i.e., $p(v,\theta) = \sum_i p_i(v,\theta)$ for every vertex v and color θ). Further, $p_1, p_2, \ldots$ are said to be *non-overlapping* if their union is feasible.

A *disk* of radius r centered at vertex v is a painting p in which the area within a radius r of v is colored by some unique color θ^v, i.e., for every vertex u,

$$p(u,\theta^v) = \begin{cases} w(u) & \text{if } d(v,u) \le r \\ 0 & \text{if } d(v,u) - w(u) \ge r \\ r - (d(v,u) - w(u)) & \text{otherwise} \end{cases}$$

A painting that comprises a union of disks centered at terminals is called a *disk painting*. This is the only kind of painting that we will use in this paper; hence, we will often simply call it a painting. A vertex u is *inside* the disk if $d(v,u)$ is *strictly* less than r, and on the *boundary* if it is not inside but has a neighbor that is inside. The *continent* of a disk is the set of vertices inside the disk.

Having described some of the key concepts of disk paintings, let us now give a high-level description of our algorithmic technique based on disk paintings. We will formally describe our algorithm later for all proper functions, but for the purpose of this informal discussion, let us focus on the SF problem. Our algorithms can be thought of as *augmented greedy* algorithms. In every online step, we initially perform a greedy augmentation of the primal solution that satisfies the new constraint, i.e., buy the cheapest path in the SF problem between the terminal pair after reducing the cost of all nodes in the current solution to 0. To account for the resulting increase in the cost of the primal solution, we aim to add a disk of radius equal to (or a constant factor of) the increase in the primal cost. If we are able to place such a disk centered at either of the two terminals in the new pair without violating feasibility of the disk painting, then we terminate the online step. The more challenging scenario is when such disks cause infeasibility of the painting. In this case, the algorithm augments the primal solution with a graph element that depends on the problem. For example, in the case of SF in general graphs, the algorithm connects the two terminals to a vertex on which the infeasibility occurred. On the other hand, for SF in graphs with an excluded minor, the algorithm connects all the centers of the intersecting disks via a *spider*. A spider is a tree with at most one vertex of degree greater than two, which is called the *center* of the spider. The paths connecting the center to the leaves are called the *legs* of the spider. The crux of the analysis is to show that the total primal cost in these instances where we are unable to add a new disk to the disk painting can be amortized to the existing disks by charging the disks that caused the infeasibility.

Note: Due to lack of space, detailed proofs are deferred to the full version of the paper.

II. Preliminaries

Before we describe the algorithms for the network design problem, we need to introduce a few notations. Consider a graph G with a vertex-weight function w and a proper function f defined over $2^{V(G)}$. Let T denote the set of vertices with zero weight. A vertex t is a *terminal* of a proper function f, if $f(\{t\}) = 1$. Note that by Terminality, $t \in T$. For a set $S \subseteq V$, let $\delta(S) \subseteq V \setminus S$ denote the neighbors of S.

Properties of Proper Functions. The following is the direct result of the disjointness of a proper function.

Proposition 2. *If a set S does not contain a terminal, then $f(S) = 0$.*

A proper function is *trivial* if for every $S \subseteq V$, $f(S) = 0$. Observe that if for a set S, $f(S) = 1$, then $f(V \setminus S) = 1$. Thus by Proposition 2, both S and $V \setminus S$ contain at least one terminal. This leads to the next proposition.

Proposition 3. *Any non-trivial proper function has at least two terminals.*

For a subset $X \subseteq V$, let $CC(X)$ denote the collection of connected components of $G[X]$. The following lemma (formally proved in [11]) gives a polynomial-time test for whether a set X is feasible. The lemma easily follows from applying the properties of proper functions on the connected components of $G[X]$.

Lemma 1 (Lemma 7 in [11])**.** *A subset X is feasible if and only if X contains all the terminals and $f(C) = 0$ for every $C \in CC(X)$.*

Lemma 1, together with the Efficiency property, guarantees that given a subset $X \subseteq V$, in polynomial time we can either (i) find a set S such that $f(S) = 1$ and $\delta(S) \cap X = \phi$; or (ii) verify that X is a feasible solution. Indeed the following lemma provides a more refined structural property of a feasible solution.

Lemma 2. *Let $S \subseteq T \subseteq V$ and let X be a feasible solution. If $f(S) = 1$ and there are no terminals in $T \setminus S$, then $G[X]$ contains a path from a terminal $\tau \in S$ to a vertex $v \in \delta(T)$.*

Lemma 2 is particularly interesting when T is the continent of a disk centered at a terminal. If the other terminals are not inside the disk, then any feasible solution connects the center to a vertex on the boundary.

Given a graph G, *contracting* a connected set of vertices S denotes replacing S by a *super-vertex* adjacent to $\delta(S)$. A *contraction* of a graph G (denoted by $\overline{G}$) is obtained by contracting connected subsets of vertices in G. For a vertex v in $\overline{G}$, let $\gamma(v)$ denote the set of vertices of G that have been contracted to form v. If a vertex v is not part of a contracted set, then it retains its label, i.e., $\gamma(v) = \{v\}$. We note that by definition, for every v in $V(\overline{G})$, $\gamma(v)$ is connected in G. We extend the notation by defining $\gamma(S) = \bigcup_{v \in S} \gamma(v)$ for any set $S \subseteq V(\overline{G})$. For simplicity, we consider G to be a contraction of itself. We refer to the original vertices in G as *simple vertices*.

Let $V(G)$ denote the set of vertices of G, and let w_G be a node weight function over $V(G)$. For a contraction $\overline{G}$, we derive a corresponding weight function by reducing the cost of super-vertices to zero, i.e., for every vertex $v \in V(\overline{G})$,

$$w_{\overline{G}}(v) = \begin{cases} w_G(v) & \text{if } \gamma(v) = \{v\} \\ 0 & \text{otherwise} \end{cases}$$

Let H be an induced subgraph of G. Let $\overline{G}$ be a contraction. The *contracted subgraph* $\overline{H}$ is obtained from H by contracting $V(H) \cap \gamma(v)$ to v for every $v \in \overline{G}$. Note that $\overline{H} \subseteq \overline{G}$. When there is no ambiguity, a contracted subgraph may retain the label of the original subgraph, i.e., we may refer to $\overline{H}$ by H as well.

Let f be a proper function w.r.t. the graph G. Given a contraction $\overline{G}$, a *contracted function* $f^{\overline{G}}$ is obtained by setting $f^{\overline{G}}(S) = f(\bigcup_{v \in S} \gamma(v))$ for every set $S \subseteq V(\overline{G})$. In other words, the cuts retain their f-value. Indeed, $f^{\overline{G}}$ is proper too.

Proposition 4. *The contracted function $f^{\overline{G}}$ is proper w.r.t. to $\overline{G}$ and $w_{\overline{G}}$.*

Properties of Disk Paintings. The following propositions hold for a set of non-overlapping disks.

Proposition 5. *If u is on the boundary of a disk centered at v, then $p(u, \theta^v)$ is strictly positive.*

Proposition 6. *A vertex inside a disk cannot be on the boundary of another disk.*

Fact 1. *Since the weight of a terminal is zero, the center of a disk is inside the disk.*

We emphasize that by Proposition 6, the disks may share a vertex *only* on their boundaries. This is indeed a crucial observation which ultimately leads to our algorithm for the network design problem. Fact 1 is implicitly used in our analysis since we assume the center of a disk is inside the disk no matter how small is the radius. The next lemma shows the relationship between a painting and the optimum offline solution for SF. This lemma is also not used explicitly in our analysis; however, we exploit it to design our algorithms. This is the key property of disk paintings which might be of independent interest.

Lemma 3. *Let $\mathscr{L}$ be a painting comprising disks centered at a subset of terminals S such that for any terminal pair (s,t), s (resp., t) is not inside a disk centered at t (resp., s). If T be any subgraph of G connecting all terminal pairs with at least one terminal in S, then $w(T)$ is at least the sum of the radii of the disks.*

III. Online Node-weighted Network Design

We start by describing a special variant of the well-studied facility location problem. The input comprises a set of facilities each with a setup cost, and a set of clients each with a connection cost to every facility, and a set of connectivity demands. The *group non-metric facility location* problem (GNFL) asks for a mapping of clients to facilities that minimizes the sum of setup costs and connection costs while satisfying *demands* defined below.

Let Λ and Ψ denote the set of facilities and clients, respectively. For a facility $\lambda \in \Lambda$, let $\omega(\lambda) \in \mathbb{R}_{\geq 0}$ denote the setup cost of λ. For a client $\psi \in \Psi$ and a facility $\lambda \in \Lambda$, let $\overline{d}(\psi, \lambda) \in \mathbb{R}_{\geq 0}$ denote the cost of connecting ψ to λ.

A mapping $M : \Psi \to \Lambda$ assigns clients to facilities. A mapping is a partial function, i.e., some of the clients may not be connected to a facility. A facility λ is *open* if for some client ψ, $M(\psi) = \lambda$. The cost $c(M)$ of a mapping M is the sum of setup costs of open facilities and connection costs used in the mapping, i.e.,

$$c(M) = \sum_{\lambda \mid \exists \psi, M(\psi) = \lambda} \omega(\lambda) + \sum_{\psi, \lambda \mid M(\psi) = \lambda} \overline{d}(\psi, \lambda)$$

In the GNFL problem, the demands are in the form of groups of clients $\mathscr{D} = \langle g_1, g_2, \ldots \rangle$ where $g_i \subseteq \Psi$. A mapping M *satisfies* $\mathscr{D}$ if for every group $g_i \in D$ at least one client $\lambda \in g_i$ is mapped to a facility. Given a set of demands, the goal is to find a mapping of minimum cost that satisfies all demands. In the online variant of the problem, the set of facilities and setup costs are known in advance but the demands arrive online one at a time, revealing the connection costs of a client if it was not present in previous demands. Upon receiving a new demand, we need to augment the mapping to cover at least one client of the new demand.

Bounded Group Non-metric Facility Location. Given a graph $G = (V, E)$ with node weights w, the bounded

group non-metric facility location problem w.r.t a real value r (r-BFL) is an instance of the GNFL problem as follows. Recall that $T \subseteq V$ is the set of zero-weight vertices.

- For every vertex $v \in V$ we have a facility (with the same label);
- The setup cost function is identical to the node-weight function;
- For every zero-weight vertex $t \in T$ we have a client (with the same label); and
- For a client $t \in T$, consider a disk of radius r centered at t in G. For every vertex v on the *boundary* of the disk, the connection cost between t and v is $\overline{d}(t,v) = d^w(t,v) - w(v)$. For every other client-facility pair (t,v) the connection cost is infinity. Note that this includes the facilities that are too far ($d^w(t,v) - w(v) \geq r$) as well as those that are too close ($d^w(t,v) < r$).

In other words, a client can be mapped only to the facilities on the boundary of the disk, the cost of which is the distance for *touching* the facility in G.

For a pair of groups of vertices g_1 and g_2, let $d^w(g_1,g_2) = \min_{u \in g_1, v \in g_2} d^w(u,v)$ be the distance between the groups in G. Let $\mathscr{D}$ denote the set of demands. We restrict the input of r-BFL by adding the following assumption.

- Every pair of demands $g_1, g_2 \in \mathscr{D}$ should be at least $2r$ far from each other in G, i.e., $d^w(g_1,g_2) \geq 2r$.

At any time in the algorithm, we say a client is *active* if it has appeared in a demand so far. When the exact radius is not a concern, we may refer to r-BFL as BFL. Given a mapping M, the graph $H(M)$ is the subgraph of G induced by the vertices of shortest paths connecting t to v for every client t and facility v such that $M(t) = v$. Observe that $w(H(M)) \leq c(M)$.

We note that since the demands are disjoint in BFL, we may collapse every group to a single client and thus it reduces to the *non-metric facility location problem (NFL)*[22], [23]. In other words, we replace clients in a demand g by a special client c_g such that for every facility v, $\overline{d}(c_g,v) = \min_{t \in g} \overline{d}(t,v)$. Let BFLALG be an online algorithm for the BFL problem with competitive ratio α_{BFL}. We use BFLALG as a black-box to show that the network design problem admits a competitive ratio of $O(\log(k) \cdot \alpha_{BFL})$. In fact, Alon et al. [23] give an online randomized algorithm for the NFL problem with competitive ratio $O(\log(k)\log(n))$, i.e., $\alpha_{BFL} = O(\log(k)\log(n))$. (Here n is the number of facilities and k is the number of active clients). Therefore our competitive ratio is $O(\log^2 k \log n)$.

Algorithm for Online Network Design. We are now ready to describe algorithm NDALG For every integer $i \in \mathbb{Z}$, the algorithm keeps a 2^i-BFL instance $\mathscr{L}_i$. We augment the mapping of the instances using BFLALG . Let $\mathscr{D}_i$ and M_i denote the demands and the current mapping for the instance $\mathscr{L}_i$, respectively. Our algorithm maintains a partial solution X guaranteeing that for every i, the solution for $\mathscr{L}_i$ is included in X, i.e., $H(M_i) \subseteq X$. Let I denote the number of BFL instances with at least one demand. Indeed one can show that the algorithm can be modified such that $I = O(\log(k))$, the details are presented in the full version of the paper.

The algorithm maintains the invariant that for any demand in $\mathscr{L}_i$ (corresponding to a group of clients g) the following *neighborhood clearance (NC)* holds at the time of arrival of the demand in $\mathscr{L}_i$.

Definition 1 (NC(g,i))**.** *The neighborhood of a group $g \subseteq T$ is* clear *in $\mathscr{L}_i$ if both the following conditions hold:*

- *The group g is at least $2 \cdot 2^i$ far in G from any previous demand in $\mathscr{D}_i$; and*
- *For every (currently) open facility v in M_i, the connection cost $\overline{d}(t,v)$ is infinite for every $t \in g$.*

If one of the conditions fails, $NC(g,i)$ does not hold and an active client of $\mathscr{L}_i$ or an open facility of $\mathscr{L}_i$ closest to g is the witness *of the failure.*

The algorithm starts by initializing X and M_i's to empty. At any time step h, let f_h be the cumulative function. Let $\mathscr{T}_h$ denote the set of terminals of f_h. We augment the solution X iteratively until it satisfies f_h. At each iteration, the following process is executed.

Let X be the current partial solution. Let $\overline{G}$ denote a contraction of G by contracting every connected component of $G[X]$. Let f denote the contracted function of f_h w.r.t. $\overline{G}$. Recall that by Proposition 3, f has at least two terminals. Let (τ_1, τ_2) denote the closest pair of terminals of f w.r.t. $w_{\overline{G}}$. Let D be the distance between them. We first buy the shortest path between τ_1 and τ_2 (thus incurring a cost of D). Note that this shortest path may contain super-vertices. Adding a super-vertex u to X implies setting X as the union of X and $\gamma(u)$. Recall that $\gamma(u)$ is the set of simple vertices contracted to u. Since $\gamma(u)$ denotes a connected component of X, adding a super-vertex u to X does not change X in this step of the algorithm.

Consider the integer i such that $4 \cdot 2^i \leq D < 4 \cdot 2^{i+1}$. Let $g(\tau_1) = \gamma(\tau_1) \cap \mathscr{T}_h$ (resp. $g(\tau_2) = \gamma(\tau_2) \cap \mathscr{T}_h$) be the group of terminals of f_h contracted to τ_1 (resp. τ_2). If the neighborhood of either $g(\tau_1)$ or $g(\tau_2)$ is clear, we give the corresponding group as a new demand to BFLALG for $\mathscr{L}_i$. We mimic the solution of BFLALG , i.e., if M_i

is augmented by mapping a client $t \in T$ to a facility $v \in V$, we buy the shortest path in G connecting t to v. However, if none of the neighborhoods is clear, let z_1 and z_2 denote the witnesses of failure corresponding to $NC(g(\tau_1), i)$ and $NC(g(\tau_2), i)$. We then connect τ_1 to z_1 and τ_2 to z_2 by buying shortest paths w.r.t. w.

Analysis. We distinguish between two types of costs incurred by the algorithm. The *simulation cost* is the total weight of vertices being purchased for covering the augmentations of mappings in every iteration, i.e., the simulation cost is at most $\sum_i w(H(M_i))$. The cost due to the weight of other vertices in X is called the *connectivity cost*. Note that we may incur a connectivity cost in two places in the algorithm: (i) when buying the shortest path; or (ii) when buying the paths connecting terminals to the corresponding witnesses of failure.

Recall that I denotes the number of BFL instances with at least one demand. Let OPT denote the weight of an optimal (offline) solution of the network design problem. We show that both simulation and connectivity costs are at most $O(I \cdot \alpha_{BFL}) \cdot OPT$. For every $i \in \mathbb{Z}$, let OPT_i denote the cost of an optimal (offline) solution for $\mathscr{L}_i$. First we show that OPT_i is a lower bound for OPT. Intuitively, by applying Lemma 2 in every iteration, one can show that for every demand $g \in \mathscr{D}_i$, the optimal solution contains a path from a terminal $t \in g$ to a vertex at the boundary of a disk of radius 2^i centered at t. Indeed such a path connects t to a facility in $\mathscr{L}_i$. Thus we can get a feasible solution for $\mathscr{L}_i$ by opening a facility at the intersection of the optimal solution with the boundaries of the disks centered at the active terminals and then connecting every demand to the closest open facility.

Lemma 4. *For every $i \in \mathbb{Z}$, $OPT_i \leq OPT$.*

We are now ready to prove the bound on the cost of the algorithm.

Lemma 5. *The total cost incurred by the algorithm is within $O(I \cdot \alpha_{BFL})$ factor of OPT.*

Lemma 4 directly leads to the desired bound for the simulation cost:

$$\sum_i w(H(M_i)) \leq \sum_i c(M_i) \leq \alpha_{BFL} \sum_i OPT_i \leq \alpha_{BFL} I \cdot OPT$$

We show a similar upper bound for the connectivity cost. For every i, let $\mathscr{D}_i$ denote the set of demands so far for $\mathscr{L}_i$. First we claim that BFLALG incurs the cost at least 2^i for satisfying each demand.

Claim 1. *For every i, $c(M_i) \geq |\mathscr{D}_i| \cdot 2^i$.*

Consider an arbitrary iteration of the algorithm. Suppose h demands have arrived so far and let f_h denote the cumulative function. Let $\mathscr{T}_h \subseteq V(G)$ denote the terminals of f_h. Let X be the partial solution at the start of iteration and let $\overline{G}$ be the contraction obtained from G by contracting the connected components of X. Let f denote the corresponding contracted function. In the algorithm, we find the closest pair of terminals (τ_1, τ_2) that are D far from each other. We buy the shortest path between τ_1 and τ_2, thus incurring a connectivity cost D.

We partition the iterations into different classes. Class i comprises iterations for which i satisfies $4 \cdot 2^i \leq D < 4 \cdot 2^{i+1}$. We show at any time in the algorithm, the total connection cost incurred for Class i iterations is bounded by $O(1) \cdot c(M_i)$ which, together with Claim 1, completes the proof of the lemma.

IV. Online Network Design in Graphs Excluding a Fixed Minor

Before we describe the deterministic algorithm for H-minor-free graphs, we need to introduce some notation. In the rest of the section, unless specified otherwise, all graphs are H-minor-free for a fixed graph H.

A painting is said to be *r-uniform* if it is a union of disks whose radii are r. When the exact radius is not important, we may refer to an r-uniform painting as a uniform painting.

Disk Fitting. We use a *disk fitting* process called DISKFIT repeatedly to construct a painting. Recall that a vertex v is simple, if $v \in V(G)$. Let $\mathscr{L}$ be a painting of a contraction $\overline{G}$. Given a connected set S of simple vertices and a radius r, the DISKFIT process tries to contract S and add a disk of radius r centered at the resulting super-vertex in $\mathscr{L}$. We call this a trial. The trial might either be successful, or fail for multiple reasons that we describe below.

If for any reason the trial finishes unsuccessfully, $\mathscr{L}$ and $\overline{G}$ will remain unchanged and the process returns a vertex called a *witness*. Let us now describe a trial. If for a vertex $v \in S$, v is already contracted in $\overline{G}$ or $\mathscr{L}(v) > 0$, the trial fails and the corresponding witness is defined as the (super-)vertex $u \in V(\overline{G})$ for which $v \in \gamma(u)$. Otherwise, the trial contracts S to a vertex s. Let $\overline{G}'$ denote the resulting graph. Note that $\mathscr{L}$ is still a valid painting for $\overline{G}'$. Let p be a disk of radius r centered at s in $\overline{G}'$. If the union of p and $\mathscr{L}$ is feasible in $\overline{G}'$, we augment $\mathscr{L}$ to $\mathscr{L} + p$ (after changing the underlying graph to $\overline{G}'$) and the trial terminates successfully. Otherwise, p and $\mathscr{L}$ are said to *intersect* at the vertices where $\mathscr{L} + p$ is infeasible. The

trial terminates unsuccessfully by reporting an infeasible vertex in $\mathscr{L}+p$ as the witness, breaking ties arbitrarily.

Binding Spiders. Let $\mathscr{L}$ be the union of a set of disks on a contraction $\overline{G}$. Suppose v is a witness reported by the DISKFIT process in an unsuccessful trial for adding a disk over a set of simple (and connected) vertices S. Let L_v denote the centers of the disks whose boundary or continent contains v. (Note that if a vertex of S is already contracted in $\overline{G}$, then the witness may be inside a disk.) A binding spider w.r.t. to the witness v reported by DISKFIT process is a spider in $\overline{G}$ centered at v and connected with shortest paths w.r.t. $w_{\overline{G}}$ to (i) every vertex in L_v, and (ii) the vertex $u \in V(\overline{G})$ with $\gamma(u) \cap S \neq \phi$ that is closest to v. The following shows that the cost of buying a binding spider depends only on the degree of the center.

Lemma 6. *Let Υ be a binding spider w.r.t. an unsuccessful trial for putting a disk of radius r, centered at a set S, in an r-uniform painting. If the center of spider has degree d, then $w_{\overline{G}}(\Upsilon) \leq d \cdot r + w(S \cap \gamma(\Upsilon))$.*

In the algorithm, we only call the DISKFIT process for a set S if S is already in the output X. Therefore Lemma 6 implies that the cost of buying a binding spider is at most $w(\gamma(\Upsilon) \setminus X) \leq w_{\overline{G}}(\Upsilon) - w(S \cap \gamma(\Upsilon)) \leq d \cdot r$.

Recall that for a fixed graph H, the average degree of an H-minor-free graph is bounded by some constant c_H. (It is shown in [24], [25] that $c_H = O(h\sqrt{\log h})$ where $h = |V(H)|$.) For simplicity of notation, we introduce two constants $\alpha = \max\{c_H, 3\}$ and $\mu = 2\alpha$. The following lemma together with Lemma 6 provides a means to charge the cost of spiders *with sufficiently large number of legs* to (the radii of) disks in our analysis. Note that if a binding spider has more than two leaves, its center has to be on the boundary of multiple disks.

Lemma 7. *Let $\mathscr{L}$ be the union of N disks on an H-minor-free graph. For a vertex v, let $\eta(v)$ denote the number of disks whose boundary contains v. Then,*

$$\sum_{v \mid \eta(v) \geq \alpha} \eta(v) \leq \alpha \cdot N.$$

Algorithm for H-Minor-Free Graphs. We are now ready to describe algorithm MFSFALG . For every $i \in \mathbb{Z}$, the algorithm keeps a painting $\mathscr{L}_i$ on a contraction of G. Throughout the algorithm, the changes to the painting $\mathscr{L}_i$ are made only by adding disks of radius 2^i using the DISKFIT process. Therefore, $\mathscr{L}_i$ is a 2^i-uniform painting comprising non-overlapping disks of radius 2^i.

Initially, $\mathscr{L}_i$'s are empty paintings on G. At a time step h, let f_h denote the cumulative function. We iteratively augment the solution until f_h is satisfied. In every iteration, let X denote the current partial solution and let $\overline{G}$ denote a contraction obtained by contracting every connected component of $G[X]$. Let f be the corresponding contracted function. By Proposition 3, there are at least two terminals in $\overline{G}$. Let (τ_1, τ_2) denote the closest pair of terminals of f. Let $D = d^{w_{\overline{G}}}(\tau_1, \tau_2)$. We first buy the shortest path in $\overline{G}$ connecting τ_1 to τ_2. Now consider the integer i such that $\mu \cdot 2^i < D \leq \mu \cdot 2^{i+1}$. Using the DISKFIT process, we try putting a disk of radius 2^i centered at $\gamma(\tau_1)$ in $\mathscr{L}_i$. If the trial is unsuccessful, we do the same process for $\gamma(\tau_2)$. If both trials are unsuccessful, let c_1 and c_2 denote the corresponding witnesses of failure. We buy the binding spiders w.r.t. c_1 and c_2.

Analysis. Let I denote the number of paintings with at least one disk. We will now show that the competitive ratio of MFSFALG is $O(I)$. Indeed, the same argument as that of Section III, shows that the algorithm can be slightly modified such that $I = O(\log k)$, thereby proving Theorem 2. Let *OPT* denote the weight of an optimal solution to the network design problem. L*et C_i denote the centers of disks in $\mathscr{L}_i$. First, we show the relationship between the paintings and *OPT*.

Lemma 8. *For every i, the total radii of disks in $\mathscr{L}_i$ is a lower bound for the optimal solution.*

We are now ready to prove the main lemma.

Lemma 9. *The total cost incurred by the algorithm is at most $O(I) \cdot OPT$.*

Let X denote the partial solution at an iteration of the algorithm. For every $v \in V(G)$, let $\tilde{w}(v)$ denote the *cost of buying* a vertex v w.r.t. X, i.e., in $\tilde{w}(v) = w(v)$ if $v \notin X$ and let $\tilde{w}(v) = 0$ if $x \in V$. Let (τ_1, τ_2) denote the closest pair of terminals. An iteration is of Type i if $\mu 2^i < D \leq \mu 2^{i+1}$ where $D = d^{w_{\overline{G}}}(\tau_1, \tau_2)$ is the length of the shortest path between τ_1 and τ_2. We show that the cost of the vertices purchased in all iterations of Type i is bounded by $O(|C_i| \cdot 2^i) = O(OPT)$. In the rest of proof, we only consider Type i iterations for an arbitrary i.

Recall that in every iteration, we buy a path connecting τ_1 and τ_2 and we may additionally buy two binding spiders. A binding spider Υ is said to be *expensive* if $\tilde{w}(\Upsilon)$, the cost of buying it, is strictly more than $\alpha \cdot 2^i$. We classify iterations as follows:

1) Process DISKFIT successfully added a disk of radius 2^i centered at either τ_1 or τ_2.
2) Process DISKFIT was unsuccessful and neither of

the binding spiders was expensive.

3) Process DISKFIT was unsuccessful and at least one binding spider was expensive.

For each of the above cases, we can show that the increase in the weight of X can be charged to the radii of the disks in $\mathscr{L}_i$. Further details are presented in the full version of the paper.

ACKNOWLEDGMENT

The authors would like to thank Anupam Gupta for suggesting that dual disks lead to simpler proofs of online edge-weighted network design results.

REFERENCES

[1] M. Imase and B. M. Waxman, "Dynamic steiner tree problem," *SIAM J. Discrete Math.*, vol. 4, no. 3, pp. 369–384, 1991.

[2] B. Awerbuch, Y. Azar, and Y. Bartal, "On-line generalized steiner problem," *Theor. Comput. Sci.*, vol. 324, no. 2-3, pp. 313–324, 2004.

[3] P. Berman and C. Coulston, "On-line algorithms for steiner tree problems (extended abstract)," in *STOC*, 1997, pp. 344–353.

[4] A. Agrawal, P. N. Klein, and R. Ravi, "When trees collide: An approximation algorithm for the generalized steiner problem on networks," *SIAM J. Comput.*, vol. 24, no. 3, pp. 440–456, 1995.

[5] M. X. Goemans and D. P. Williamson, "A general approximation technique for constrained forest problems," *SIAM J. Comput.*, vol. 24, no. 2, pp. 296–317, 1995.

[6] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor, "The online set cover problem," *SIAM J. Comput.*, vol. 39, no. 2, pp. 361–370, 2009.

[7] J. Naor, D. Panigrahi, and M. Singh, "Online node-weighted steiner tree and related problems," in *FOCS*, 2011, pp. 210–219.

[8] M. Charikar, C. Chekuri, T.-Y. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li, "Approximation algorithms for directed steiner problems," *J. Algorithms*, vol. 33, no. 1, pp. 73–91, 1999.

[9] P. N. Klein and R. Ravi, "A nearly best-possible approximation algorithm for node-weighted steiner trees," *J. Algorithms*, vol. 19, no. 1, pp. 104–115, 1995.

[10] S. Guha and S. Khuller, "Improved methods for approximating node weighted steiner trees and connected dominating sets," *Inf. Comput.*, vol. 150, no. 1, pp. 57–74, 1999.

[11] E. D. Demaine, M. Hajiaghayi, and P. N. Klein, "Node-weighted steiner tree and group steiner tree in planar graphs," in *ICALP (1)*, 2009, pp. 328–340.

[12] C. Moldenhauer, "Primal-dual approximation algorithms for node-weighted steiner forest on planar graphs," *Inf. Comput.*, vol. 222, pp. 293–306, 2013.

[13] C. Chekuri, A. Ene, and A. Vakilian, "Node-weighted network design in planar and minor-closed families of graphs," in *ICALP (1)*, 2012, pp. 206–217.

[14] S. Guha, A. Moss, J. Naor, and B. Schieber, "Efficient recovery from power outage (extended abstract)," in *STOC*, 1999, pp. 574–582.

[15] A. Moss and Y. Rabani, "Approximation algorithms for constrained node weighted steiner tree problems," *SIAM J. Comput.*, vol. 37, no. 2, pp. 460–481, 2007.

[16] Z. Nutov, "Approximating steiner networks with node-weights," *SIAM J. Comput.*, vol. 39, no. 7, pp. 3001–3022, 2010.

[17] J. Qian and D. P. Williamson, "An O(log$n$)-competitive algorithm for online constrained forest problems," in *ICALP (1)*, 2011, pp. 37–48.

[18] C.-L. Li, S. T. McCormick, and D. Simchi-Levi, "The point-to-point delivery and connection problems: complexity and algorithms," *Discrete Applied Mathematics*, vol. 36, no. 3, pp. 267–292, 1992.

[19] J. Edmonds and E. L. Johnson, "Matching: A well-solved class of integer linear programs," in *Combinatorial Optimization*, 2001, pp. 27–30.

[20] S. Korman, "On the use of randomization in the online set cover problem," *M.S. thesis, Weizmann Institute of Science*, 2005.

[21] N. Buchbinder and J. Naor, "The design of competitive online algorithms via a primal-dual approach," *Foundations and Trends in Theoretical Computer Science*, vol. 3, no. 2-3, pp. 93–263, 2009.

[22] D. S. Hochbaum, "Heuristics for the fixed cost median problem," *Mathematical programming*, vol. 22, no. 1, pp. 148–162, 1982.

[23] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor, "A general approach to online network optimization problems," *ACM Transactions on Algorithms*, vol. 2, no. 4, pp. 640–660, 2006.

[24] W. Mader, "Homomorphieeigenschaften und mittlere kantendichte von graphen," *Mathematische Annalen*, vol. 174, no. 4, pp. 265–268, 1967.

[25] A. V. Kostochka, "Lower bound of the Hadwiger number of graphs by their average degree," *Combinatorica*, vol. 4, no. 4, pp. 307–316, 1984.