Online Probabilistic Metric Embedding: A General Framework for Bypassing Inherent Bounds

Yair Bartal* Nova Fandina† Seeun William Umboh‡

January 6, 2020

Abstract

Probabilistic metric embedding into trees is a powerful technique for designing online algorithms. The standard approach is to embed the entire underlying metric into a tree metric and then solve the problem on the latter. The overhead in the competitive ratio depends on the expected distortion of the embedding, which is logarithmic in n, the size of the underlying metric. For many online applications, such as online network design problems, it is natural to ask if it is possible to construct such embeddings in an online fashion such that the distortion would be a polylogarithmic function of k, the number of terminals.

Our first main contribution is answering this question negatively, exhibiting a *lower bound* of $\tilde{\Omega}(\log k \log \Phi)$, where Φ is the aspect ratio of the set of terminals, showing that a simple modification of the probabilistic embedding into trees of Bartal (FOCS 1996), which has expected distortion of $O(\log k \log \Phi)$, is *nearly-tight*. Unfortunately, this may result in a very bad (polynomial) dependence in terms of k.

Our second main contribution is a general framework for bypassing this limitation. We show that for a large class of online problems this online probabilistic embedding can still be used to devise an algorithm with $O(\min\{\log k \log(k\lambda), \log^3 k\})$ overhead in the competitive ratio, where k is the current number of terminals, and λ is a measure of subadditivity of the cost function, which is at most r, the current number of requests. In particular, this implies the first algorithms with competitive ratio polylog(k) for online *subadditive network design* (*buy-at-bulk network design* being a special case), and polylog(k, r) for online *group Steiner forest*.

*Hebrew University of Jerusalem, Israel. `yair@cs.huji.ac.il` Supported in part by ISF grant 1817/17.

†Hebrew University of Jerusalem, Israel. `fandina@cs.huji.ac.il` Supported in part by ISF grant 1817/17 and by the Leibnitz Foundation.

‡The University of Sydney, Australia. `william.umboh@sydney.edu.au` Supported in part by NWO grant 639.022.211 and ISF grant 1817/17. Part of this work was done while a postdoc at Eindhoven University of Technology, and while visiting the Hebrew University of Jerusalem.

1 Introduction

Low-distortion metric embeddings play an important role in the design of algorithms. Many applications are based on reducing the problem on arbitrary metrics to instances defined over a simple metric, where we may be able to devise efficient approximation or online algorithms, while incurring only a small loss in quality of the solution depending on the distortion of the embedding.

One of the most successful methods has been probabilistic embedding into ultrametrics [Bar96] where an arbitrary metric space is embedded probabilistically into special tree metrics (HSTs, or ultrametrics). The main theorem provides a tight $O(\log n)$ bound on the distortion [Bar04, FRT04], where n is the size of the metric. This result has been used in a plethora of applications in various areas, including clustering [BCR01], metric labeling [KT02], network design [AA97, GKR98], linear arrangement and spreading metrics [Bar04], among others. See [Ind01] for more applications.

The goal of this paper is to deepen the study of metric embedding and its applications in the online setting. An *online algorithm* receives requests one by one and needs to satisfy each request immediately without knowing future requests. Previously, the dominant approach in applying metric embeddings to online algorithms is to use an offline embedding of the entire underlying metric (V, d_V) into a tree metric or HST using the probabilistic embeddings of [Bar04, FRT04] and then solving the problem there. This approach has been particularly useful for online algorithms, with many applications including subadditive network design [AA97], group Steiner tree/forest [AAA$^+$06, NPS11], metrical task system and k-server problems [BBBT97, BBMN15, BCL$^+$18], reordering buffers [ERW10], and file migration [Bar96], among others.

There are two main disadvantages to this approach: The first is that it requires upfront knowledge of the underlying metric; The second, perhaps more crucial, is the inherent $O(\log n)$ overhead incurred in the resulting competitive ratio bound, adding a dependence on n, which may be unrelated to the natural parameters of the problem (e.g., in terms of the problem's lower bounds). This first issue, i.e. the necessity of knowing the underlying metric, is due to the use of the offline embeddings of [Bar04, FRT04]. This has motivated several researchers, including Indyk et al. [IMSZ10], to initiate the study of *online embeddings*. In this setting, the embedding algorithm is given the points of the metric one by one and needs to produce consistent metric embeddings at each stage. They observed that the probabilistic embedding method of Bartal [Bar96] can be quite simply implemented in an online fashion to achieve the distortion bound of $O(\log \hat{k} \log \hat{\Phi})$, where $\hat{k}$ is the *final* number of requested points, and $\hat{\Phi}$ is the aspect ratio of the underlying metric. However, while this alternative approach (partially) overcomes the first issue, it fails to address the second: First, it still requires pre-knowledge of $\hat{k}$—the final number of requested points—and second, it introduces an additional dependence on another parameter, the aspect ratio, which may in general be exponential in $\hat{k}$, and by itself is often also not a natural parameter of the problem.

This paper makes two main contributions: First, we show that the aspect ratio is a *necessary* parameter for any strictly online embedding, more specifically showing that the above bound cannot be much improved except for removing the necessity to know the parameters in advance. The second main contribution is providing a general framework for overcoming this obstacle in order to use such embeddings to obtain general bounds for online problems based on algorithms for trees with an overhead that is polylogarithmic in k, the number of terminal points which have appeared in requests up to the current stage. We expect further extensions and applications of this approach to arise in the near future.

Online embedding model. An online embedding algorithm is given an underlying metric space

(V, d_V). Then, it receives a sequence of points (called *terminals*) one by one; when a point arrives, the algorithm embeds it into a destination metric space (M, d_M) while committing to previous decisions. More formally, an online embedding is defined as follows.

Definition 1.1 (Online embedding)**.** *An* online embedding f *of a sequence of terminal points* $x_1, \ldots, x_k$ *from a metric space* (V, d_V) *into a metric space* (M, d_M) *is a sequence of embeddings* $f_1, \ldots, f_k$ *such that for each* $1 \leq i \leq k$*:* ***(1)*** f_i *is a non-contractive embedding of* $x_1, \ldots, x_i$ *into* M*,* ***(2)*** f_i extends *the previous embedding* f_{i-1}*, i.e. for each* $x \in \{x_1, \ldots, x_{i-1}\}$*,* $f_i(x) = f_{i-1}(x)$*. The* distortion *of the online embedding* f *is the distortion of the final embedding* f_k*. A* probabilistic online embedding *is a probability distribution over online embeddings.*

Note that the results of [Bar04, FRT04] imply that there exists a probabilistic online embedding into hierarchically separated trees with expected distortion $O(\log n)$, where $n = |V|$. A *hierarchically separated tree* (HST) is a rooted tree T where each edge has a non-negative length and the edge lengths decrease geometrically along every root-to-leaf path.

Fully extendable embeddings. A key application of online embeddings is to reduce an online problem defined over an arbitrary metric (V, d_V) to one defined over a "simpler" metric (M, d_M). For this to work, we should be able to translate a feasible solution obtained on (M, d_M), which in general may use Steiner points (points in M that are not images of the terminals) to a feasible solution in (V, d_V). Moreover, for the analysis of the reduction, we should be able to translate an optimal solution in (V, d_V) into a feasible solution in (M, d_M). Here, it may definitely be the case that Steiner points in V, i.e. non-terminal points, are used in an optimal solution in V. We therefore further require the online embedding to be *fully extendable*, that is both f and f^{-1} must have appropriate metric extensions.

Definition 1.2 (Fully extendable embeddings)**.** *Let* $f = (f_1, \ldots, f_k)$ *be an online embedding of a sequence of terminals* $x_1, \ldots, x_k \in V$ *into* M*. For all* $1 \leq i \leq k$*, let* α_i *denote the distortion of* f_i*. We say that* f *is* fully extendable *to* V *with respect to* $\{M_i\}_{i \leq k}$*,* $M_1 \subseteq M_2 \ldots \subseteq M_k \subseteq M$*, if for all* i *there exist* $F_i \colon V \to M$ *and* $H_i \colon M_i \to V$ *such that*[1]*:* ***(i)*** F_i *and* H_i *depend only on terminals* $\{x_1, \ldots, x_i\}$*;* ***(ii)*** F_i *extends* f_i *and for all* $u \neq v \in V$*,* $d_M(F_i(u), F_i(v)) \leq \alpha_i \cdot d_V(u, v)$*;* ***(iii)*** H_i *extends* f_i^{-1} *and for all* $\hat{u} \neq \hat{v} \in M_i$*,* $d_V(H_i(\hat{u}), H_i(\hat{v})) \leq d_M(\hat{u}, \hat{v})$*;* ***(iv)*** H_{i+1} *extends* H_i*. A probabilistic online embedding is fully extendable if for each embedding* f *in its support, there exist extensions* F_i *and* H_i *such that properties (i), (iii), and (iv) hold for each embedding, and property (ii) holds in expectation, i.e. for all* $u \neq v \in V$*,* $E_{F_i}[d_M(F_i(u), F_i(v))] \leq \alpha_i \cdot d_V(u, v)$*, where* α_i *is the expected distortion of the probabilistic embedding* f_i*.*

Roughly speaking, one should think of the non-terminal points in M_i as Steiner points that are useful in defining M_i. For example, if f_i embeds the terminals into the leaves of a tree T_i, then $M_i = T_i$.

1.1 Our Results

We begin by showing that a somewhat more delicate application of the approach of Bartal [Bar96] yields a probabilistic online embedding into HSTs with expected distortion $O(\log k \log \Phi)$ which does not require knowledge of the underlying metric space.

[1] The inverse of an injective function should be interpreted for its restriction to its image.

Theorem 1.3. *For any metric space* (V, d_V) *and sequence of terminals* $x_1, \ldots, x_k \in V$*, there exists a probabilistic online embedding into HSTs with expected distortion* $O(\log k \log \Phi)$*, where* Φ *is the aspect ratio of the terminals.*[2]

Our first main result is a lower bound on the distortion of every probabilistic online embedding into trees, showing that the above bound is nearly tight. We also prove a better lower bound for probabilistic online embeddings into HSTs.

Theorem 1.4. *There exists an infinite family of metric spaces* $\{(V_\ell, d_{V_\ell})\}$ *and terminal sequences* $\{\sigma_\ell\}$ *such that every probabilistic online embedding of* σ_ℓ *into trees has expected distortion* $\Omega(\log k \log \Phi_\ell / (\log\log k + \log\log \Phi_\ell))$*, where* $k = |\sigma_\ell|$ *and* Φ_ℓ *is the aspect ratio of* (V_ℓ, d_{V_ℓ}) *which can be as large as* $2^{k^{(1-\delta)/2}}$ *for any fixed* $\delta \in (0, 1)$*.*

Theorem 1.5. *There exists an infinite family of metric spaces* $\{(V_\ell, d_{V_\ell})\}$ *and terminal sequences* $\{\sigma_\ell\}$ *such that every probabilistic online embedding of* σ_ℓ *into HSTs has expected distortion at least* $\Omega(\log k \log \Phi_\ell / \log\log k)$*, where* $k = |\sigma_\ell|$ *and* Φ_ℓ *is the aspect ratio of* (V_ℓ, d_{V_ℓ}) *which can be as large as* $2^{k^{1-\delta}}$ *for any fixed* $\delta \in (0, 1)$*.*

This is almost-tight and also implies that as a function of k, a polynomial dependence is required.

Our second main result is that despite the lower bound of Theorem 1.5, which forces a dependence of Φ on the embedding distortion, for a broad class of online problems, it is still possible to use the HST embedding of Theorem 1.3 in a more clever manner such that the overhead of using the embedding is only $O(\min\{\log k \log(kr), \log^3 k\})$, where r is the number of requests. This is much less than the distortion of the embedding, effectively bypassing the distortion lower bound of Theorem 1.5. We call this class of problems *abstract network design*. It captures problems on metric spaces that are amenable to the usual tree embedding approach, including well-known network design problems such as group Steiner forest and buy-at-bulk network design, and other online problems involving metrics such as s-server[3], reordering buffer, and distributed paging.

Abstract network design. The basic idea behind the definition is to capture problems on graphs that are amenable to the usual tree-embedding-based approach, and whose feasible solutions can be defined in terms of a subset of vertices, called terminals.

In an instance of abstract network design, the algorithm is given a connected graph $G = (V, E)$ with edge lengths $d : E \to \mathbb{R}_{\geq 0}$. At each time step i, the algorithm is given a request that consists of a set of terminals $Z_i \subseteq V$. Let $\mathcal{Z}_i = \cup_{j \leq i} Z_j$, the set of terminals seen so far. The algorithm has to respond with a response $\mathcal{R}_i = (R_i, C_i)$, where $R_i \subseteq E$ is a subgraph of G, and a *connectivity list* C_i which is an ordered subset of terminal pairs from $\binom{\mathcal{Z}_i}{2}$. A solution to the first i requests is a sequence of responses $\mathcal{S}_i := (\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_i)$. The algorithm is also given at each time step a *feasibility function* $\mathcal{F}_i = \mathcal{F}_i[Z_1, Z_2, \ldots, Z_i]$ mapping a sequence of connectivity lists $(C_1, C_2, \ldots, C_i)$ to $\{0, 1\}$. The solution $\mathcal{S}_i$ is *feasible* iff $\mathcal{F}_i(C_1, C_2, \ldots, C_i) = 1$ and every pair in C_j is connected in R_j for each $j \leq i$. A valid algorithm must maintain feasible solutions $\mathcal{S}_1, \mathcal{S}_2, \ldots \mathcal{S}_r$.

A feasibility function is called *memoryless* if it obeys that whenever $\mathcal{F}_i(C_1, C_2, \ldots, C_i) = 1$, then for all $(C'_1, C'_2, \ldots, C'_{i-1})$ such that $\mathcal{F}_{i-1}(C'_1, C'_2, \ldots, C'_{i-1}) = 1$, it holds that $\mathcal{F}_i(C'_1, C'_2, \ldots, C'_{i-1}, C_i) =$

[2] Note that this can be made into $O(k \log k)$. Essentially, we only need to maintain a probabilistic partition for each "relevant" distance scale and there can be at most $\min\{k, \log \Phi\}$ of these.

[3] This is usually called k-server but we use s to denote the number of servers to avoid confusion with our use of k.

1. E.g., most standard network design problems, such as Steiner tree, possess a memoryless feasibility function.

Next, we specify how the cost of a solution $\mathcal{S}$ is determined. To include a wide class of problems, we use the general framework of subadditive functions. In particular, the cost of using an edge e is its length $d(e)$ times a subadditive function of the subset I of time steps i in which $e \in R_i$. At each time step i, the algorithm is given oracle access to a *load function* $\rho_i : 2^{\{1,\ldots,i\}} \to \mathbb{R}_{\geq 0}$ that is *subadditive*[4], monotone-increasing and satisfies $\rho_i(I) = 0$ if and only if $I = \emptyset$. We also require that ρ_i extends ρ_{i-1} for each i. To simplify notation, for a set I of time steps, we write $\rho(I) = \rho_j(I)$ where j is the latest time step in I. The cost of a solution $\mathcal{S}_i$ is $\text{cost}(\mathcal{S}_i) = \sum_{e \in E} d(e)\rho(\{j \leq i : e \in R_j\})$. The goal is to minimize the cost. When the algorithm is randomized, we measure the expected cost of its solution. We also define the parameter $\lambda_\rho = \min\{\max_I \rho(I)/\min_{I \neq \emptyset} \rho(I), r\}$ which is, intuitively, a measure of the level of subadditivity of the function.

We denote the number of requests in the online problem by r and write $\mathcal{Z} = \mathcal{Z}_r$, and we use k to denote the total number of terminal points $|\mathcal{Z}_r|$.

It is often convenient to assume w.l.o.g. (the proof is in Appendix A) that G is a complete graph, with edge lengths satisfying triangle inequality.

For the sake of concreteness, we demonstrate how to express four of our applications in our framework. The first three problems share the same load function ρ which satisfies $\rho(I) = 1$ when $I \neq \emptyset$ and $\rho(\emptyset) = 0$; note that $\lambda_\rho = 1$.

Example 1 (Steiner Forest)**.** *In the online Steiner Forest problem, a sequence of vertex pairs (s_i, t_i) arrives online, and the goal is to maintain a subgraph G' such that each (s_i, t_i) that has arrived so far is connected in G'. The cost of a solution G' is the total length of its edges.*

In the Abstract Network Design formulation, the terminal set of the i-th request consists of s_i and t_i. The feasibility function $\mathcal{F}_i$ is defined by $\mathcal{F}_i(C_1, \ldots, C_i) = 1$ if and only if $C_j = \{(s_j, t_j)\}$ for every $1 \leq j \leq i$.

Example 2 (Constrained Forest)**.** *In the online Constrained Forest problem [GW95], each request consists of a set of terminals Z_i and a* cut requirement function *$g_i : 2^{Z_i} \to \{0, 1\}$ that is* proper*: $g_i(\emptyset) = g_i(Z_i) = 0$, $g_i(X) = g_i(Z_i - X)$ for all $X \subseteq Z_i$ and $g_i(X \cup Y) \leq \max\{g_i(X), g_i(Y)\}$ for all disjoint sets $X, Y \subseteq Z_i$. The goal is to maintain a subgraph G' that satisfies each request (Z_i, g_i) seen so far, i.e. for every vertex subset $S \subseteq V$ such that $g_i(S \cap Z_i) = 1$, there is at least one edge in G' that leaves the set S. The cost of G' is the total length of its edges.*

In the Abstract Network Design formulation, the terminal set of the i-th request is Z_i. The feasibility function $\mathcal{F}_i$ is defined by $\mathcal{F}_i(C_1, \ldots, C_i) = 1$ if and only if for every $1 \leq j \leq i$, for every terminal subset $S \subseteq Z_j$ such that $g_j(S \cap Z_j) = 1$, there is at least one edge in C_j that leaves the set S.

Example 3 (Group Steiner Forest)**.** *In the online Group Steiner Forest problem, each request consists of a pair of vertex sets (S_i, T_i). The goal is to maintain a subgraph G' such that for each request (S_i, T_i) seen so far, we have that G' connects some vertex in S_i to some vertex in T_i. The cost of G' is the total length of its edges.*

In the Abstract Network Design formulation, the terminal set of the i-th request is $S_i \cup T_i$. The feasibility function $\mathcal{F}_i$ is defined by $\mathcal{F}_i(C_1, \ldots, C_i) = 1$ if and only if for every $1 \leq j \leq i$, we have that $C_i = \{(s_i, t_i)\}$ for some $s_i \in S_i$ and some $t_i \in T_i$.

[4]For any pair of sets $A, B \subseteq \{1, \ldots, i\}$, it holds $\rho_i(A \cup B) \leq \rho_i(A) + \rho_i(B)$.

Example 4 (s-server)**.** *In the online s-server problem, we are given initial locations of s servers. Each request consists of a single vertex t_i. The goal is to move a server from its current location to t_i, which incurs a cost equal to the distance travelled.*

In the Abstract Network Design formulation, the terminal set of the i-th request consists of the set of points on which the first i requests of the s-server instance appeared on and the initial locations of the s servers. The feasibility function $\mathcal{F}_i$ is defined by $\mathcal{F}_i(C_1, \ldots, C_i) = 1$ if for each $j \le i$, C_j is a single pair (u, v) (which corresponds to moving a server from u to a request location v) and the sequence of $(C_1, \ldots, C_i)$ corresponds to a valid movement of the servers. The load function ρ is simply the cardinality function since we pay $d(e)$ each time e is traversed; here $\lambda_\rho = r$.

Note that the feasibility functions for the online network design problems are memoryless. On the other hand, the s-server problem is not memoryless since we need to check that $(C_1, \ldots, C_i)$ corresponds to a valid movement of servers.

Our second main result is that for abstract network design problems satisfying some mild conditions, it is possible to use the online embedding of Theorem 1.3 such that the overhead due to the embedding is only polylog in k and $\lambda_\rho \le r$. The two ingredients that we need is a "baseline" algorithm on general metrics with a finite competitive ratio β (that can be arbitrarily large, possibly dependent on r and k), and a scheme for combining any two algorithms A and B to give a "combined" algorithm C such that for any instance, the cost of C is not much more than the minimum of the cost of A and B. We say that online problems that has such a baseline algorithm and combining scheme admits a "min operator".

Definition 1.6 (Min operator)**.** *An online problem admits a* min operator *with factor $\eta \ge 1$ if it has a competitive algorithm (possibly randomized), and for any two deterministic online algorithms A and B, there is a deterministic online algorithm C that on every instance of the problem satisfies $\text{cost}(C) \le \eta \cdot \min\{\text{cost}(A), \text{cost}(B)\}$, where $\text{cost}(\cdot)$ is the cost of the respective algorithm. Moreover, if either A or B is randomized, then C is also randomized and has $E[\text{cost}(C)] \le \eta \cdot \min\{E[\text{cost}(A)], E[\text{cost}(B)]\}$. If $\eta = O(1)$, we simply say that the problem admits a min operator.*

Note that the definition for the case when A and B are deterministic implies the definition for the case when at least one of them is randomized since a randomized algorithm is a distribution of deterministic algorithms.

Theorem 1.7. *Consider an abstract network design problem. If it admits a min operator and if there exists an algorithm that is α-competitive on instances where the input graph is an HST then there exists a randomized algorithm that, on every instance, has expected competitive ratio $O(\alpha \cdot \min\{\log k \cdot \log(k\alpha\lambda_\rho), \log^2 k \cdot \log(k\alpha)\})$.*

Applications: Network design problems. Our model naturally captures online network design problems and their generalizations to subadditive costs. As mentioned earlier, these problems have memoryless feasibility functions. In Appendix B (Corollary B.5), we show that such problems admit a min operator. We now state applications to two network design problems: Constrained Forest, and Group Steiner Forest.

It is easy to see that the Constrained Forest problem can be solved exactly on trees, even with subadditive costs. Thus, we can apply Theorem 1.7 with $\alpha = 1$ to get:

Corollary 1.8. *There is a randomized $O(\min\{\log k \cdot \log(k\lambda_\rho), \log^3 k\})$-competitive algorithm for the Subadditive Constrained Forest problem.*

For the Group Steiner Forest problem, we restrict our attention to the load function ρ where $\rho(I) = 0$ if $I = \emptyset$ and $\rho(I) = 1$ otherwise, i.e. we simply want to minimize the total length of edges used; note that $\lambda_\rho = 1$. Naor, Panigrahi and Singh [NPS11] gave an $O(\log^4 k \log r)$-competitive algorithm on HSTs, which yields an $O(\log n \log^4 k \log r)$-competitive algorithm on general graphs when combined with the usual $O(\log n)$-distortion embedding into HSTs. Applying Theorem 1.7 with $\lambda_\rho = 1$ and $\alpha = \log^4 k \log r$ gives us:

Corollary 1.9. *There is a randomized $\tilde{O}(\log^6 k \cdot \log r)$-competitive algorithm for the Group Steiner Forest problem.*

Even though our techniques apply to any subadditive load function, there is no known algorithm that can handle general subadditive costs with a polylogarithmic competitive ratio, even on trees.

Other online metric problems. We consider several more applications of Theorem 1.7. As all these problems are Metrical Task Systems, they admit a min operator ([ABM93]), implying the applicability of the theorem.

Corollary 1.10. *There is a randomized algorithm for the distributed paging problem with competitive ratio $O(\log^4 m)$, where m is the current number of different pages requested.*

Corollary 1.11. *There is a randomized algorithm for the s-server problem with competitive ratio $O(\log^2 s \log r(\log r + \log\log s))$.*

Corollary 1.12. *There is a randomized algorithm for the reordering buffer problem with competitive ratio $O(\log b \log r(\log r + \log\log b))$, where b is the buffer size.*

Corollaries 1.10, 1.11, 1.12 follow from applying Theorem 1.7 with the value of α (the randomized competitive ratio for HSTs) being $O(\log m)$ [Bar96], $O(\log^2 s)$ [BCL+18], and $O(\log b)$ [ER17] respectively. The existing results hold for the case where the metric is known in advance where the overhead is $O(\log n)$. For s-server, there is an alternative approach providing $O(\log^4 s)$ overhead ([BCL+18, Lee18]). Our bounds provide an improvement when these overheads are large.

1.2 Our Techniques

We briefly summarize our main technical contributions here.

Upper bound for online embedding. Our online embedding is based on the construction of Bartal [Bar96]. The key idea is to use probabilistic partitions with a dynamic padding parameter, which can be bounded by $O(\log k)$ using the analyses of [Bar04, ABN06]. Achieving dependence on the current number of terminals k rather than the final number of requested points $\hat{k}$ is made possible via a technique of [ABN06], where the probabilistic partitions are used in the construction with padding parameter polynomial in k. We maintain a hierarchical partition of the terminals, one per distance scale, such that for every scale, the partition at that scale is a refinement of the higher-scale partitions. A technical challenge is maintaining the refinement property of the partitions as new distance scales appear over time causing an increase in the aspect ratio Φ. We then show that our construction can be modified to be fully extendable (see Section 4 for a discussion), which is necessary for its applications.

Lower bounds. We use a recursive construction to build the underlying graph. The idea is to use as a base graph a high-girth expander which has many long edge-disjoint paths. The existence of such a graph follows from the probabilistic method and the results of Alon and Capalbo [AC07] on packing edge-disjoint paths in expanders. Applying Yao's principle, we use the underlying graph to define a distribution over terminal sequences and vertex pairs such that any deterministic online embedding into a tree has expected distortion $\Omega(\ell \log k)$, where ℓ is the number of recursive levels.

Abstract network design. To prove Theorem 1.7, we augment the usual application of tree embeddings to online problems. Consider an abstract network design problem that admits a min operator and an algorithm for HST instances. Our approach uses the combining scheme to combine the usual tree-embedding-based algorithm and the baseline algorithm. At a high level, the scheme allows us to fall back on the baseline algorithm in the case that the tree solution becomes very expensive due to some graph edges being distorted badly. Essentially, this lets us replace the logarithmic dependency on Φ with a logarithmic dependency on the competitive ratio of the baseline algorithm, the number of terminals k and λ_ρ (Theorem 6.1). Since the dependency on the competitive ratio of the baseline algorithm decreases exponentially, this idea can be applied repeatedly to completely remove the dependency on the baseline algorithm as well.

1.3 Other Related Work

The only previous work that tries to go beyond the $O(\log n)$-distortion overhead of the usual tree embedding approach are the recent results of Bubeck et al. [BCL+18] and Lee [Lee18] for the s-server problem. The former gave a reduction to HSTs with an overhead of $O(\log s \log \Phi)$ where Φ is the aspect ratio of the entire underlying metric space. Lee exhibits a sophisticated dynamic embedding into HSTs which, together with the randomized online algorithm for the s-server problem on HSTs of Bubeck et al. [BCL+18], provides a randomized online algorithm for general metrics whose competitive ratio is polylogarithmic in s. The dynamic embedding contributes a $O(\log^2 s \cdot c(s)) = O(\log^4 s)$ overhead to the $c(s) = O(\log^2 s)$ competitive ratio achieved by Bubeck et al. on HSTs. However, this dynamic embedding result is somewhat ad-hoc, tailored particularly to the s-server problem and the analysis approach of Bubeck et al.

Previous work on online embeddings. The notion of online embedding was previously considered by [IMSZ10]. They considered a more restrictive model in which each f_i only depends on the terminals seen so far $x_1, \ldots, x_i$; in comparison our model allows the algorithm to use information of the whole underlying metric space V when constructing f_i. (For instance, embedding the entire underlying metric is allowed in our model but not theirs.) We call their model *metric-oblivious* and ours *non-metric-oblivious*. For our upper bounds, we will actually construct a metric-oblivious embedding, while we will prove lower bounds for non-metric-oblivious embeddings.

While the work of Lee [Lee18] and ours share the same overarching goal of obtaining an overhead of the embedding approach that depends solely on the number of "relevant" points, the two works deal with very different settings. In our case, we are interested in the set of terminal points, the points appearing in requests so far, which grows with the number of requests, and our goal is to provide quite general methods for dealing with such situations. In comparison, in Lee's work the set of points of interest is of fixed size s, but changes dynamically, and the result is tailored to apply to the type of algorithms and analysis arising within the specific randomized (or fractional) s-server setting.

Previous work on subadditive network design. In the offline setting, the only results on general subadditive network design that we are aware of is an $O(\log k)$-approximation by combining the Awerbuch-Azar technique with a simple modification of the tree embedding of [FRT04] due to [GNR10]. (See proof of Lemma 6.10) Most of the literature on subadditive network design focuses on the well-studied buy-at-bulk case where the subadditive function f is of the form $f(A) = g(|A|)$ for some concave function g. For the buy-at-bulk problem, the best approximation is achieved by the same $O(\log k)$-approximation algorithm for the general subadditive problem and there is a hardness result of $O(\log^{1/4-\epsilon} k)$ [And04]; in the single-sink setting, where the sinks t_i are equal, many $O(1)$-approximations are known, e.g. [GKK$^+$01, Tal02, GKR04, GI06, GMM09, JR09, GR10, GP12].

In the online setting, the only work on the general subadditive network design problem is due to [GHR06]. Assuming that the underlying metric is known to the algorithm, a derandomization of oblivious network design from [GHR06] gives an $O(\log^2 n)$-competitive algorithm. As in the offline setting, most of the previous work is on the buy-at-bulk problem. For the special cases of the Steiner tree and generalized Steiner forest problems, tight deterministic $O(\log k)$-competitive algorithms are known [IW91, BC97]. There are also deterministic $O(\log k)$-competitive algorithms for the online rent-or-buy problem [AAB04, BCI01, Umb15]. For the buy-at-bulk problem, the only prior work is on the single-sink case: [GRTU17] gave a deterministic $O(\log k)$-competitive algorithm and also observed that is possible to use online tree embeddings and obtain a randomized $O(\log^2 k)$-competitive algorithm.

1.4 Organization of the Paper

Section 2 covers basic notation and terminology used in the rest of the paper. In Section 3, we discuss how to use a fully extendable online embedding from (V, d_V) to (M, d_M) to (approximately) reduce an abstract network design problem defined over the former to one defined over the latter. Then, we give an overview of our online embedding construction in Section 4 (details are in Appendix C. Next, we describe our lower bound construction in Section 5. Finally, we present our online algorithm for abstract network design problems that admit a min operator in Section 6.

2 Preliminaries

Notation and terminology. The *distortion* of an embedding $f : V \to Y$ is its maximum expansion times maximum contraction, i.e. $\mathit{distortion}(f) = \max_{u \neq v \in V} \frac{d_Y(f(u), f(v))}{d_V(u,v)} \cdot \max_{u \neq v \in V} \frac{d_V(u,v)}{d_Y((f(u)),(f(v)))}$. We say that the embedding is *non-contractive* if for all $u \neq v \in V$, $d_Y(f(u), f(v)) \geq d_V(u, v)$ and *non-expansive* if for all $u \neq v \in V$, $d_Y(f(u), f(v)) \leq d_V(u, v)$ The *aspect ratio* of V is $\Phi(V) = d_{\max}(V)/d_{\min}(V)$ where $d_{\max}(V) = \max_{u \neq v \in V} d_V(u, v)$ and $d_{\min}(V) = \min_{u \neq v \in V} d_V(u, v)$.

Hierarchically separated trees (HSTs). HST metrics were defined in [Bar96]:

Definition 2.1. *A μ-HST metric is the shortest path metric defined on the leaves of a weighted tree T that satisfies the following: (1) the edge weight from any node to each of its children is the same, and (2) the edge weights along any path from the root to any leaf are decreasing by a factor of at least μ.*

The following definition is equivalent to the one given above (up to a constant), we use these representations interchangeably through the paper:

Definition 2.2. *A μ-HST metric is the metric defined on the leaves of a rooted tree T with the following properties. Each node v of T has an associated label $\Delta(v) \geq 0$, such that $\Delta(v) = 0$ iff v is a leaf, and for any two nodes $u \neq v$, if v is a child of u then $\Delta(v) \leq \Delta(u)/\mu$. The distance between two leaves $u \neq v$ is given by $d_T(u, v) = \Delta(lca(u, v))$, where $lca(u, v)$ is the least common ancestor of u and v.*

3 Using Online Embeddings for Abstract Network Design

In this section, we show how to use a fully extendable online embedding f from (V, d_V) to (M, d_M) to reduce an instance of an abstract network design problem P defined over the former to one defined over the latter, with an overhead equal to the distortion of f.

Definition 3.1 (Instance induced by embedding)**.** *Consider an instance of P defined on (V, d_V) with request sequence $Z_1, \ldots, Z_r \subseteq V$, feasibility functions $\mathcal{F}_1, \ldots, \mathcal{F}_r$ and load function ρ. Let $f = (f_1, \ldots, f_r)$ be an online embedding of the request sequence into a tree metric (T, d_T) and $T_i = T[\mathcal{Z}_i]$. Then, f* induces *the following instance of P on (T, d_T): the sequence of requests of the induced instance is $f(Z_1), \ldots, f(Z_r) \subseteq M$; given a sequence of connectivity lists $C'_i \subseteq \binom{f(\mathcal{Z}_i)}{2}$, its i-th feasibility function $\mathcal{F}'_i$ satisfies $\mathcal{F}'_i(C'_1, \ldots, C'_i) = \mathcal{F}_i(f^{-1}(C'_1), \ldots, f^{-1}(C'_i))$; and it has the same load function ρ as the original instance.*

In Section 3.1, we describe how to use fully extendable online embeddings into tree metrics, which will be needed for the proof of Theorem 1.7 in Section 6. In Section 3.2, we describe how to generalize the approach to work with fully extendable online embeddings into other families of metrics.

3.1 Using Online Embeddings into Trees

For an embedding into a tree metric, it will be useful to have extension H_i of f_i^{-1} to the subtree induced by the terminals seen so far.

Definition 3.2 (Fully extendable online tree embedding)**.** *Consider an online embedding $f = (f_1, \ldots, f_k)$ of a sequence of terminal points $x_1, \ldots, x_k$ from (V, d_V) into a tree metric (T, d_T). Let T_i be the subtree of T induced by $f_i(\{x_1, \ldots, x_i\})$. Then, we say that is* fully extendable *if it is fully extendable with respect to $\{T_i\}_{i \leq k}$. For a probabilistic embedding, we require that every tree embedding in its support is fully extendable.*

We say that an online tree embedding algorithm is fully extendable *if given any input metric space (V, d_V) and any online sequence of terminal points from (V, d_V), the algorithm produces a fully extendable online embedding of the terminal sequence into a tree metric.*

A crucial property of instances on tree metrics is that the algorithm only needs to consider the subtree induced by the set of terminals seen so far.

Proposition 3.3. *Let P be an online Abstract Network Design problem. Consider an instance of P on a tree metric (T, d_T) with request sequence $Z_1, \ldots Z_r$, and let $T_i = T[\mathcal{Z}_i]$ where $\mathcal{Z}_i = Z_1 \cup \ldots \cup Z_i$. Then, for any feasible solution $\mathcal{S} = ((R_1, C_1), \ldots, (R_r, C_r))$, there exists a feasible solution $\mathcal{S}' = ((R'_1, C_1), \ldots, (R'_r, C_r))$ such that each R'_i is contained in T_i and* $\text{cost}(\mathcal{S}') \leq \text{cost}(\mathcal{S})$.

Proof. Consider the solution $\mathcal{S}'$ with $R'_i = \bigcup_{(u,v)\in C_i} p_T(u,v)$, where $p_T(u,v)$ is the unique path in T between u and v. Clearly, $\mathcal{S}'$ is a feasible solution. We also have that $R'_i \subseteq T_i \cap R_i$. Thus, since the load function is monotone, we get $\text{cost}(\mathcal{S}') \leq \text{cost}(\mathcal{S})$. □

We are now ready to prove the following reduction theorem.

Theorem 3.4. *Let P be an online Abstract Network Design problem. Suppose that there exists an online algorithm $\mathsf{Alg}^{\mathcal{T}}$ for P over tree metrics with competitive ratio β, and that there exists a fully extendable online tree embedding algorithm $\mathsf{Embed}^{\mathcal{T}}$ with distortion α. Then, there exists an online algorithm Alg for P over arbitrary metrics with competitive ratio $\alpha\beta$. Moreover, when either $\mathsf{Embed}^{\mathcal{T}}$ or $\mathsf{Alg}^{\mathcal{T}}$ is randomized, then Alg is randomized with expected competitive ratio $\alpha\beta$.*

Proof. For simplicity, we only prove the case when both $\mathsf{Embed}^{\mathcal{T}}$ and $\mathsf{Alg}^{\mathcal{T}}$ are deterministic; the proof for the randomized case is similar. The algorithm Alg works as follows. For each request Z_i and feasibility function $\mathcal{F}_i$, the algorithm Alg uses the online embedding algorithm $\mathsf{Embed}^{\mathcal{T}}$ to compute the embedding f_i of $\mathcal{Z}_i$, and it feeds the request $f(Z_i)$ and feasibility function $\mathcal{F}'_i$ to $\mathsf{Alg}^{\mathcal{T}}$. Let (R_i^T, C_i^T) be the response of $\mathsf{Alg}^{\mathcal{T}}$; note that $R_i^T \subseteq T_i$ by Proposition 3.3. Then, the algorithm Alg constructs its response (R_i, C_i) by translating $\mathsf{Alg}^{\mathcal{T}}$'s response to V using the extension function H_i of f^{-1}, i.e. $R_i = H_i(R_i^T)$ and $C_i = f_i^{-1}(C_i^T)$. Observe that $x, y \in T$ are connected in R_i^T if and only if $H_i(x), H_i(y) \in V$ is connected in R_i. Together with the fact that H_i extends f_i^{-1}, we get that every pair in C_i is connected in R_i. By feasibility of $\mathsf{Alg}^{\mathcal{T}}$, we also have $\mathcal{F}_i(C_1, \ldots, C_i) = 1$. Thus, the solution $\mathcal{S}_i = ((R_1, C_1), \ldots, (R_i, C_i))$ maintained by algorithm Alg is feasible.

Finally, we analyze the competitive ratio of algorithm Alg. Let $\mathcal{S}^* = ((R_1^*, C_1^*), \ldots, (R_r^*, C_r^*))$ denote an optimal solution for the instance on V and OPT be its cost. Also, let $\text{OPT}(T)$ denote the cost of an optimal solution on T.

Claim 3.5. $\text{cost}(\mathcal{S}) \leq \beta\, \text{OPT}(T)$.

Proof. For $u, v \in V$, define $I_{u,v} = \{i : (u,v) \in R_i\}$ and similarly, for $(x,y) \in T$, define $I_{x,y}^T = \{i : (x,y) \in R_i^T\}$. For brevity, we write $H = H_r$. Since H extends H_i for every $i < r$, we have $R_i = H_i(R_i^T) = H(R_i^T)$ and so $(u,v) \in R_i$ if and only if there exists $(x,y) \in R_i^T$ such that $H(\{x,y\}) = \{u,v\}$. Thus, $I_{u,v} = \bigcup_{(x,y)\in T: H(\{x,y\})=\{u,v\}} I_{x,y}^T$. Using subadditivity of ρ, we upper bound $\text{cost}(\mathcal{S})$ by $\text{cost}(\mathcal{S}^T)$ as follows:

$$\begin{aligned}
\text{cost}(\mathcal{S}) &= \sum_{u,v\in V} d_V(u,v)\rho(I_{u,v}) \\
&\leq \sum_{u,v\in V} d_V(u,v) \sum_{(x,y)\in T: H(\{x,y\})=\{u,v\}} \rho(I_{x,y}^T) \\
&= \sum_{(x,y)\in T} d_V(H(x), H(y))\rho(I_{x,y}^T) \\
&\leq \sum_{(x,y)\in T} d_T(x,y)\rho(I_{x,y}^T) = \text{cost}(\mathcal{S}^T),
\end{aligned}$$

where the last inequality is due to the non-expansiveness of H. Since $\mathsf{Alg}^{\mathcal{T}}$ is β-competitive on tree metrics, we have that $\text{cost}(\mathcal{S}) \leq \text{cost}(\mathcal{S}^T) \leq \beta\, \text{OPT}(T)$. □

Claim 3.6. $\text{OPT}(T) \leq \alpha\, \text{OPT}$.

Proof. Consider the solution on T formed by embedding $\mathcal{S}^*$ into T using the extension function F of f, i.e. its responses are $(\widehat{R}_i, \widehat{C}_i)$ where $\widehat{R}_i = F(R_i^*)$ and $\widehat{C}_i = F(C_i^*)$. The embedded solution is feasible for the instance on T and has cost at most $\sum_{u,v \in V} d_T(u,v)\rho(I^*_{u,v}) \leq \alpha \sum_{u,v\in V} d_V(u,v)\rho(I^*_{u,v}) = \alpha\,\mathrm{OPT}$ because f has distortion at most α. ☐

With these claims in hand, we conclude that Alg is $\alpha\beta$-competitive. ☐

3.2 Using Online Embeddings into Other Families of Metrics

The main property of tree metrics that the proof of Theorem 3.4 uses is Proposition 3.3 which enables the algorithm ALG to translate the solution on the tree $\mathcal{S}^T$ back into the original metric. While the proposition is not true for any arbitrary family $\mathcal{M}$ of metrics, we can still apply the same approach by requiring that the online embedding algorithm $\mathsf{Embed}^{\mathcal{M}}$ and the online algorithm $\mathsf{Alg}^{\mathcal{M}}$ for P on metrics in $\mathcal{M}$ satisfy the following additional properties.

Consider an instance of P on a metric (V, d_V) with request sequence $Z_1, \ldots, Z_r$. Let f be an online embedding of $Z_1, \ldots, Z_r$ into a metric (M, d_M) that is fully extendable with respect to $M_1 \subseteq \ldots \subseteq M_r \subseteq M$, and $\mathcal{S}^M = ((R_1^M, C_1^M), \ldots, (R_r^M, C_r^M))$ be a solution to the instance on (M, d_M) induced by f. Then, f and $\mathcal{S}^M$ are *compatible* if M_i contains the vertex set of the subgraph R_i^M for each $1 \leq i \leq r$. When either f or $\mathcal{S}^M$ is probabilistic, then we require that the above property holds for every embedding in the support of f and every solution in the support of $\mathcal{S}^M$.

Suppose $\mathcal{M}$ is a family of metrics. Let $\mathsf{Embed}^{\mathcal{M}}$ be an online embedding algorithm that embeds into $\mathcal{M}$, and $\mathsf{Alg}^{\mathcal{M}}$ be an online algorithm for P on instances over metrics belonging to $\mathcal{M}$, then $\mathsf{Embed}^{\mathcal{M}}$ and $\mathsf{Alg}^{\mathcal{M}}$ are compatible if for every instance of P on an arbitrary metric (V, d_V), algorithm A produces an online embedding f of the request sequence into a metric $(M, d_M) \in \mathcal{M}$ such that the solution $\mathcal{S}^M$ of $\mathsf{Alg}^{\mathcal{M}}$ on the instance induced by f on (M, d_M) is compatible with f.

Theorem 3.7. *Let P be an online Abstract Network Design problem and $\mathcal{M}$ be a family of metrics. Suppose that there exists an online algorithm $\mathsf{Alg}^{\mathcal{M}}$ for P over metrics belonging to $\mathcal{M}$ with competitive ratio β, and that there exists an online embedding algorithm $\mathsf{Embed}^{\mathcal{M}}$ that embeds into $\mathcal{M}$ with distortion α. Then, there exists an online algorithm* ALG *for P over arbitrary metrics with competitive ratio $\alpha\beta$. Moreover, when either $\mathsf{Embed}^{\mathcal{M}}$ or $\mathsf{Alg}^{\mathcal{M}}$ is randomized, then* ALG *is randomized with expected competitive ratio $\alpha\beta$.*

The proof of the theorem is similar to that of Theorem 3.4.

4 Online Probabilistic Embedding: Overview

We give a sketch of the construction of the online embedding of Theorem 1.3 and summarize in Theorem 4.1 the properties needed for Section 6. The full details of the construction and its analysis appear in Appendix C.

Constructing the online embedding. A Δ-*bounded probabilistic partition* of V is a distribution over partitions P of V, with cluster diameters bounded by Δ. A Δ-bounded probabilistic partition has *padding parameter* γ if for each $v \in V$ and any $\delta > 0$ the probability that $B(v, \delta\Delta/\gamma)$ is cut by the cluster of P that contains v is at most δ.

Let $X_k \subseteq V$ be the set of k terminals revealed to the online embedding thus far. We construct a collection of nested probabilistic partitions of X_k, with diameters decreasing by μ. The number of such partitions is $O(\log_\mu \Phi(X_k))$, namely, these partitions capture all scales of distances in X_k up to a factor μ. Moreover, each probabilistic partition in this collection has padding parameter $O(\log k)$. The whole hierarchical structure is maintained online. Bartal showed in [Bar96] that constructing such a hierarchical probabilistic partition of X_k implies an embedding of X_k into a distribution of HST's, with expected distortion $O(\mu \log_\mu \Phi(X_k) \log k)$.

The online Algorithm 1 in Appendix C maintains a Δ-bounded probabilistic partition of the current terminal set X_k, for a given scale Δ, with padding parameter $O(\log k)$. Notably, the algorithm does not assume the knowledge of k upfront, as is the case in previous works. The construction is based on the (offline) probabilistic partitions of [ABN06] (Lemma 5 in their paper). Their algorithm iteratively partitions a given metric space V in the following way: At the j-th step, a still unclustered point v_j is chosen in a particular way[5] that is associated with a parameter $\chi_j \geq 2$; Then, the radius r_j is randomly chosen from the distribution $p(r) = \frac{\chi_j^2}{1-\chi_j^{-2}} \frac{8 \ln \chi_j}{\Delta} \chi_j^{-\frac{8r}{\Delta}}$, for $r \in [\Delta/4, \Delta/2]$. The j-th cluster is defined to be $B(v_j, r_j)$ intersected with the still unclustered points in V. They show (Lemma 5 in [ABN06]) that this construction gives a probabilistic partition with padding parameter $O(\max_{1 \leq j \leq t} \log \chi_j)$, where t is the number of clusters that can be obtained by this construction. However, their analysis actually implies that any partition in which v_j is chosen *arbitrarily* from all the uncovered points, and r_j is chosen as before but with χ_j being *some* parameter such that $\sum_{1 \leq j \leq t} 1/\chi_j \leq 1$, has the same bound on its padding parameter (see Lemma C.1).

Algorithm 1 maintains a Δ-bounded online probabilistic partition for X_k as follows. When a new terminal x_k arrives, it is added to the first cluster by order of construction that contains x_k. Note that this is the cluster that would have contained x_k if the space V was given to the algorithm upfront. If no such cluster exists, a new cluster is created by randomly picking r according to $p(r)$ with $\chi_k = 2k^2$. The choice of χ_k is such that the sum of $1/\chi_j$ over the k steps of the algorithm is at most 1, implying the padding parameter $O(\log k)$.

We apply this procedure for each distance scale Δ to produce an online hierarchical probabilistic partition H for X_k. At each step, we maintain the number of scales $\Theta(\log_\mu \Phi(X_k))$. When a new terminal x_k arrives, the algorithm checks whether the aspect ratio of the current space has increased by at least a constant factor of μ with respect to the current top or bottom scale. If so, it adds the new top or bottom scale to H in order to maintain a valid hierarchical partition. We apply the single-scale procedure above to insert x_k in all the levels from top to bottom. The probabilistic hierarchical partitions are used to define an associated HST tree T_k.

Claim C.7 shows that our online probabilistic metric-oblivious embedding f into HST T is fully extendable. The extension F_k is obtained by mapping each non terminal $u \in V$ either into an image of one of the terminals or to the special leaf ℓ_k, which is added as a child of the root of T_k. For the extension H_k, each internal node in T is recursively mapped to the image of f_k^{-1} of one of its children.

We conclude with a summary of the properties of our embedding that are used in Section 6.

Theorem 4.1. *For any metric space (V, d_V), sequence of terminals $x_1, \ldots, x_k \in V$, and parameter $\mu > 1$, there exists a fully extendable probabilistic online embedding into a random μ-HST T such that* ***(i)*** *T has $O(k)$ edges;* ***(ii)*** *for every $u, v \in V$, and $L > 0$, we have $\Pr_T[d_T(u,v) \geq L] \leq O(\log k) \frac{d(u,v)}{L}$.*

[5] Minimizing the local growth rate, see [ABN06] for the details.

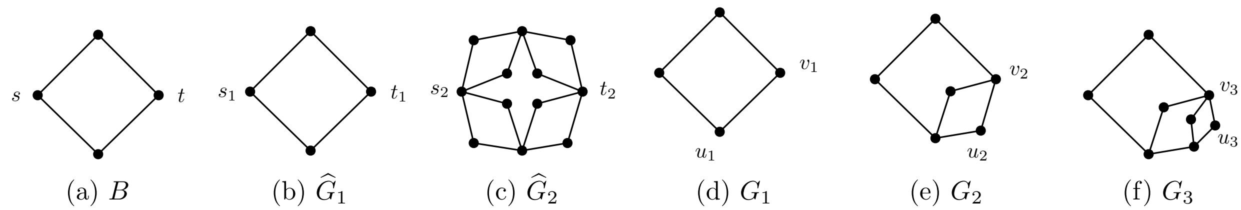

Figure 1: Example of $\widehat{G}_\ell$ and G_ℓ with a 4-cycle as base graph B. Here, $\phi = 2$ so each edge of $\widehat{G}_\ell$ has length $2^{-\ell}$.

5 Lower Bounds on Online Embedding into Trees

In this section we prove lower bounds on the expected distortion of probabilistic online embeddings into trees (Theorem 1.4). Using the standard approach for proving lower bounds against probabilistic constructions, it suffices to construct an underlying metric space (V_ℓ, d_{V_ℓ}) together with a distribution $\mathcal{D}_\ell$ of terminal sequences and vertex pairs such that for any deterministic online embedding into any HST (general tree) T, we have that $E_{(\sigma,(u,v))\in\mathcal{D}_\ell}[d_T(u,v)/d_{V_\ell}(u,v)]$ satisfies the desired lower bound.

Both lower bounds follow the same general framework, which we now describe.

5.1 Lower Bound Framework

A key ingredient is the notion of a base graph.

Definition 5.1 (Base graph)**.** *A base graph B of width ϕ is an unweighted connected graph with distinguished vertices s, t and is such that every (s,t)-path in the graph has exactly ϕ edges.*

Constructing the underlying metric space. The metric space (V_ℓ, d_{V_ℓ}) will be the shortest-path metric of a weighted graph $\widehat{G}_\ell = (V_\ell, E_\ell)$ which is constructed recursively using some suitable base graph B. Each graph $\widehat{G}_\ell$ will consist of a source vertex s_ℓ and a sink vertex t_ℓ. For the base case, the graph $\widehat{G}_1 = B$ and $s_1 = s$ and $t_1 = t$. The graph $\widehat{G}_\ell$ is constructed by taking a copy of B and then replacing each edge of B by a scaled-down copy of $\widehat{G}_{\ell-1}$ in which each edge has length $\phi^{-\ell}$. More precisely, for each edge (u,v) of B, we remove (u,v), add a scaled-down copy of $\widehat{G}_{\ell-1}$ in which each edge has length $\phi^{-\ell}$, and contract u with the copy's source and v with the copy's sink.

Constructing $\mathcal{D}_\ell$. The distribution $\mathcal{D}_\ell$ is obtained from a sequence of random edge-weighted graphs $\{G_\ell\}$. The graphs G_ℓ are defined inductively as follows: $G_1 = B$; G_ℓ is constructed by taking a copy of B, choosing a uniformly random edge (u,v) of B and replacing it with a scaled-down copy of $G_{\ell-1}$ in which each edge has its length scaled down by a factor of ϕ. See Figure 1 for an illustration of $\widehat{G}_\ell$ and G_ℓ. For each ℓ, the random graph G_ℓ defines $\mathcal{D}_\ell$ in the following way: the sequence of $\mathcal{D}_1$ consists of the vertices of G_1 in an arbitrary order and the distribution of vertex pairs of $\mathcal{D}_1$ is the uniform distribution over the edges of G_1. Inductively, the sequence of $\mathcal{D}_\ell$ consists of the vertices of B in an arbitrary order followed by the sequence of $\mathcal{D}_{\ell-1}$ corresponding to the copy of $G_{\ell-1}$ in G_ℓ. Note that the distribution $\mathcal{D}_{\ell-1}$ depends on the random choice of the edge of B that was replaced by $G_{\ell-1}$ made in the construction of G_ℓ. The distribution of vertex pairs of $\mathcal{D}_\ell$ is given by the distribution of vertex pairs of the copy of $G_{\ell-1}$ given by $\mathcal{D}_{\ell-1}$.

Constructing the base graph. We will need the following result of Rabinovich and Raz [RR98]:

Theorem 5.2 ([RR98]). *Any embedding of the n-vertex cycle into a tree has distortion at least $n/3 - 1$.*

Lemma 5.3. *Let $G = (V, E)$ be an unweighted n-vertex graph with girth g. Consider a non-contractive embedding of G into a tree T. If we choose an edge e in E uniformly at random, then* $\Pr[d_T(e) \geq ((g/3) - 1)d_V(e)] \geq (|E| - (n-1))/|E|$.

Proof. Let $C = \{e \in E : d_T(e) \geq ((g/3) - 1) \cdot d_V(e)\}$. Since we pick an edge $e \in E$ uniformly at random, we have that $\Pr[d_T(e) \geq ((g/3) - 1) \cdot d_V(e)] = |C|/|E|$. On the one hand, Theorem 5.2 implies that C intersects all cycles of G—i.e. C is a feedback edge set—and so $E \setminus C$ has no cycles; thus, $|E| - |C| \leq n - 1$. Therefore, $|C|/|E| \geq (|E| - (n-1))/|E|$, as desired. □

Lemma 5.4. *For some constant $c > 0$, there exists an infinite family $\{B_m\}$ of graphs that satisfies the following properties: **(1)** B_m is a base graph with m vertices and $O(m)$ edges; **(2)** it has width $\phi_m = \Theta(\log m)$; and **(3)** for any non-contractive embedding into a tree T, if we choose an edge e in B uniformly at random, then* $\Pr[d_T(e) \geq c \cdot \phi_m \cdot d_{B_m}(e)] \geq 1/2$.

For the proof of the lemma we will need the following:

Definition 5.5 (Very strong expanders). *An n-vertex, p-regular graph $G = (V, E)$ is a* very strong expander *if it satisfies the following:*

1. *The average degree of any subgraph of G on at most $n/10$ vertices is at most $p/6$.*
2. *The average degree of any subgraph of G on at most $n/2$ vertices is at most $2p/3$.*

We say that a collection of vertex pairs $(s_1, t_1), \ldots, (s_r, t_r)$ is *disjoint* if each vertex is chosen as s_i or t_i at most once.

The existence of an infinite family of very strong expanders with constant degree and logarithmic girth can be proved using either the probabilistic method or Ramanujan graphs [LPS88].

Lemma 5.6. *There exists an infinite family $\{G_n\}$ of graphs where G_n are very strong p-regular expanders on n vertices, for some constant $p > 0$, and has girth $\Omega(\log n)$.*

Theorem 5.7 ([AC07]). *Let $G = (V, E)$ be a very strong p-regular expander on n vertices. Then, G contains edge-disjoint paths for any $r = np/(150 \log n)$ disjoint vertex pairs $(s_1, t_1), \ldots, (s_r, t_r)$.*

We now proceed with the proof of Lemma 5.4.

Proof of Lemma 5.4. Let $G_n = (V_n, E_n)$ be a graph from the family guaranteed by Lemma 5.6. Since G_n is a p-regular, there exist $n/2$ disjoint vertex pairs such that the shortest path between each pair has at least $(\log n)/10$ edges. Choose $r = np/(150 \log n)$ such pairs and let $\mathcal{P}$ be the r edge-disjoint paths guaranteed by Theorem 5.7. Since these paths are edge-disjoint and $|E_n| = np/2$, we have that at least $r/2$ of these paths have at most $150 \log n$ edges. Let $P_1, \ldots, P_{r'}$ be $r' = r/2$ of these paths and s_i, t_i be the endpoints of P_i.

To obtain a base graph from G_n, we first remove edges that do not lie on any path P_i, i.e. the resulting graph is $P_1 \cup \cdots \cup P_{r'}$. Then, we add vertices s and t. For each i, we want to connect s to s_i and t to t_i. But the paths P_i may have different lengths so we cannot just directly connect s to s_i

and t to t_i. Since $|P_i| \le 150 \log n$, we will add a new path Q_i, of length at least $\log n$ connecting s to s_i, and a new path R_i, of length at least $\log n$, connecting t to t_i so that the resulting (s,t)-path has exactly $152 \log n$ edges. Note that the final graph still has girth $\Omega(\log n)$ (since we connected the vertex s to every s_i and the vertex t with every t_i with paths of lengths at least $\log n$ each), and the degree is at most p. Thus, the final graph B_m is a base graph with m vertices where $n \le m \le np$. This is because of the following: observe that $|E(B_m)| \le mp/2$ since B_m has degree at most p; on the other hand, $|E(B_m)| = r' \cdot 152 \log n \ge np/2$ edges so $mp/2 \ge |E(B_m)| = np/2$ and thus $m \ge n$. We have $m \le np$ because $m \le |E(B_m)| \le np$. Since each (s,t)-path has exactly $152 \log n = \Theta(\log m)$ edges, B_m has width $\phi_m = \Theta(\log m)$.

Finally, the last property follows from Lemma 5.3 and the fact that B_m has at least $2m$ edges and girth $\Omega(\log n) = \Omega(\log m) = \Omega(\phi_m)$. □

The following lemma says that we can choose edges with larger distortion but at smaller probability. It will be useful for proving the distortion lower bound for tree embeddings.

Lemma 5.8. *For every positive integer $t \ge 2$, and some constants $c' > 0$, there exists an infinite family $\{B'_m(t)\}$ of graphs that satisfies the following properties: **(1)** $B'_m(t)$ is a base graph with m vertices; **(2)** it has width $\phi'_m = \Theta(t \log(m/t))$; and **(3)** for any non-contractive embedding into a tree T, if we choose an edge e in $B'_m(t)$ uniformly at random, then $\Pr[d_T(e) \ge c' \cdot \phi'_m \cdot d_{B'_m(t)}(e)] \ge 1/2t$.*

Proof. Take B_n , with $n = \lceil m/t \rceil$, and replace each edge with a path of t edges. This results in a base graph $B'_m(t)$ on $m = n + |E(B_n)|(t-1)$ vertices and $|E(B'_m(t))| = t|E(B_n)|$ edges. It also has width $\phi'_m(t) = t\phi_n = \Theta(t \log(m/t))$ and girth $\Omega(t \log m)$. Applying Lemma 5.3 and the fact that

$$\frac{|E(B'_m(t))| - m}{|E(B'_m(t))|} = \frac{t|E(B_n)| - n - |E(B_n)|(t-1)}{t|E(B_n)|} = \frac{1}{t} \cdot \frac{|E(B_n)| - n}{|E(B_n)|} \ge \frac{1}{2t}.$$

where the last inequality was argued in the proof of Lemma 5.4. □

5.2 Lower Bound for Tree Embeddings

For each $i \le \ell$, let (u_i, v_i) be the random edge chosen at level i of the recursive construction of G_ℓ and (u_ℓ, v_ℓ) be the random vertex pair given by $\mathcal{D}_\ell$.

Lemma 5.9. *Consider the weighted graph $G_\ell = (V_\ell, E_\ell)$, and distribution $\mathcal{D}_\ell$ obtained from using $B'_m(t)$ as the base graph in the above framework. Then, we have $E_{(\sigma,(u_\ell,v_\ell)) \in \mathcal{D}_\ell}\left[\frac{d_T(u_\ell,v_\ell)}{d_{V_\ell}(u_\ell,v_\ell)}\right] \ge c'\phi'_m \cdot \ell \frac{1}{2t}\left(1 - \frac{1}{2t}\right)^\ell$.*

Proof. For each $i \le \ell$, define the random variable $c_i = \frac{d_T(u_i,v_i)}{d_V(u_i,v_i)}$ and let $C_i = E[c_i]$. We will need the following claim which says that conditioned on the distortion of (u_i, v_i) being at least α, the expected distortion of (u_ℓ, v_ℓ) is also at least α.

Claim 5.10. *For any $i < \ell$ and α, we have $E[c_\ell \mid c_i \ge \alpha] \ge \alpha$.*

Proof. We prove the claim by induction on ℓ. For the base case ($\ell = 1$), we want to prove that conditioned on $d_T(s,t) \ge \alpha d_V(s,t)$, then $E_{(u,v)\in G_1}[d_T(u,v)] \ge \alpha$ where the expectation is over a uniform distribution of edges of G_1. Observe that G_1 consists of a collection of shortest (s,t)-paths P_i that are edge-disjoint. Thus, it suffices to prove for each P_i, the above statement for a uniform distribution of edges of P_i, i.e. that $\frac{\sum_{e \in P_i} d_T(e)}{|P_i|} \geq \alpha$. To see why this is true, by triangle inequality, we have $\sum_{e \in P_i} d_T(e) \geq d_T(s,t) \geq \alpha d_V(s,t) = \alpha |P_i|$ since P_i is a shortest (s,t)-path. Thus, $\frac{\sum_{e \in P_i} d_T(e)}{|P_i|} \geq \alpha$, as desired.

We now prove the inductive case. Assume that $E[c_{\ell-1} \mid c_i \geq \alpha] \geq \alpha$ for all $i < \ell - 1$. Again, observe that G_ℓ consists of a collection of shortest (s_ℓ, t_ℓ)-paths P_i, so we can apply a similar argument as in the base case. □

Define $p_i = \Pr[c_i \geq c' \phi'_m]$. Let i^* be the first i such that $c_i \geq c' \phi'_m$. Observe that $\Pr[i = i^*] = p_i \prod_{1 \leq j < i}(1 - p_j)$ and $i = i^*$ implies that $c_i \geq c' \phi'_m$. Applying the above claim, we get that $C_\ell = \sum_{i=1}^{\ell} E[c_\ell \mid i = i^*] \Pr[i = i^*] \geq c' \phi'_m \sum_{i=1}^{\ell} p_i \prod_{1 \leq j < i}(1 - p_j)$. Lemma 5.8 part 3 implies that $1/2t \leq p_i \leq 1$ for each i. Given this constraint, the sum $\sum_{i=1}^{\ell} p_i \prod_{1 \leq j < i}(1 - p_j)$ is minimized by setting $p_i = 1/2t$ for each i. So,

$$c' \phi'_m \sum_{i=1}^{\ell} p_i \prod_{1 \leq j < i} (1 - p_j) \geq c' \phi'_m \sum_{i=1}^{\ell} \frac{1}{2t} \left(1 - \frac{1}{2t}\right)^{i-1} \geq c' \phi'_m \cdot \ell \frac{1}{2t} \left(1 - \frac{1}{2t}\right)^{\ell},$$

as desired. □

Proof of Theorem 1.4: By Yao's principle, Lemma 5.9 yields a sequence of $k = 2 + (m-2)\ell$ terminals such that any probabilistic online embedding into a tree has expected distortion at least $c' \phi'_m \cdot \ell \frac{1}{2t} \left(1 - \frac{1}{2t}\right)^{\ell} \geq \Omega(\log \frac{m}{t} \cdot \ell \left(1 - \frac{1}{2t}\right)^{\ell})$. Now, the aspect ratio of these terminals is $\Phi = (\phi'_m)^{\ell}$ so $\ell = \frac{\log \Phi}{\log \phi'_m} = \Omega(\frac{\log \Phi}{\log t + \log\log(m/t)})$ where the last equality is because $\phi'_m = \Theta(\log m)$. Setting $t = \ell$, we get that the expected distortion is at least $\Omega(\log \frac{m}{t} \cdot \ell \left(1 - \frac{1}{2t}\right)^{\ell}) = \Omega(\ell \log \frac{m}{\ell}) = \Omega(\frac{\log(m/\ell) \log \Phi}{\log \ell + \log\log(m/\ell)}) = \Omega(\frac{\log(m/\ell) \log \Phi}{\log\log \Phi + \log\log(m/\ell)})$ where the last inequality is because $\ell \leq \log \Phi$. Now, setting $m = \Theta(k^{(1+\delta)/2})$ and $\ell = \Theta(k^{(1-\delta)/2})$, we get that $\log(m/\ell) = \Theta(\delta \log k)$ and $\Phi \leq 2^{\Theta(\ell)}$ and this completes the proof of the theorem. □

5.3 Lower bound for HST embeddings

In the rest of this section, it will be convenient to use the labeled tree representation of HSTs. The main property of HST embeddings that our proof exploits is the following.

Proposition 5.11. *Let $G = (V, E)$ be an edge-weighted graph and consider a non-contractive embedding of its shortest-path metric (V, d_V) into an HST T. Then, for every $u, v \in V$, the subset of edges $e \in E$ with $d_T(e) \geq d_T(u,v)$ is a cut-set for u and v.*

Proof. Let $L = d_T(u,v)$. The HST T has a corresponding hierarchical partition of the metric space (V, d_V). Let P be the highest-level partition in which u and v are separated. We have that every edge $e \in E$ that is cut by P has $d_T(e) \geq L$. Since u and v are separated by P, we get that every (u,v)-path must have at least one edge that is cut by P. Therefore, the subset of edges $e \in E$ with $d_T(e) \geq d_T(u,v)$ is a cut-set for u and v. □

Lemma 5.12. *Consider the graph G_ℓ, and distribution of terminal sequences σ_ℓ and distribution of vertex pairs ρ_ℓ obtained from using B_m as a base graph in the above framework. Then, we have* $E_{\sigma_\ell, e_\ell \in \rho_\ell}\left[\frac{d_T(e_\ell)}{d_{V_\ell}(e_\ell)}\right] \geq \Omega(\ell \phi_m) = \Omega(\ell \log m)$.

Proof. Let the random variable H_i denote the support of (u_i, v_i) and define $C_{i,j} = \{e \in H_i : d_T(e) = \phi_m^{-j+1}\}$.

We begin with two observations. Firstly, $\Pr[(u_j, v_j) \in C_{j,j}] \geq 1/2$. This is because of Lemma 5.4 part 3. Secondly, if $d_T(u_{i-1}, v_{i-1}) \geq \phi_m^{-j+1}$ then Proposition 5.11 implies that in the subgraph H_i, the set of edges $C_{i,j}$ is a cut-set for u_{i-1} and v_{i-1}). Thus, conditioned on $d_T(u_{i-1}, v_{i-1}) \geq \phi_m^{-j+1}$, the probability that a random edge of H_i belongs to $C_{i,j}$ is at least the size of the min (u_{i-1}, v_{i-1})-cut divided by $|E(H_i)|$. Since every (u_{i-1}, v_{i-1})-path in H_i has exactly ϕ_m edges, we get that this ratio is at least $1/\phi_m$ so $\Pr[e_i \in C_{i,j} \mid e_{i-1} \in C_{i-1,j}] \geq 1/\phi_m$.

We have $E[d_T(e_\ell)] = \sum_{j=1}^{\ell} E[d_T(e_\ell) \mid e_j \in C_{j,j}] \cdot \Pr[e_j \in C_{j,j}]$. So, $E[d_T(e_\ell) \mid e_j \in C_{j,j}] \geq \phi_m^{-j+1} \Pr[e_\ell \in C_{\ell,j} \mid e_j \in C_{j,j}] \geq \phi_m^{-j+1} \prod_{j<i\leq \ell} \Pr[e_i \in C_{i,j} \mid e_{i-1} \in C_{i-1,j}] \geq \phi_m^{-\ell+1}$.

Putting the above together, we get $E[d_T(e_\ell)] \geq \sum_{j=1}^{\ell} \phi_m^{-\ell+1} \cdot \frac{1}{2} = \frac{\ell \phi_m}{2} d_V(e_\ell)$, as $d_V(e_\ell) = \phi_m^{-\ell}$. □

We are now ready to prove Theorem 1.5.

Proof of Theorem 1.5: Suppose we set $\ell = k^{1-\delta}$ and $m = k^{\delta}$. Then, we get that the length of the terminal sequence σ_ℓ is $k = 2 + (m-2)\ell = \Theta(m\ell)$. Moreover, the aspect ratio of the terminals is $\Phi = \phi_m^{\ell+1} = (\log m)^{\ell+1} = 2^{\Theta(\ell)}$. Thus, we get a lower bound of $\Omega(\ell \phi_m) \geq \Omega(\delta \log \Phi_\ell \log k / \log\log k)$, as desired. □

6 A General Framework for Bypassing the Lower Bound

In this section we prove Theorem 1.7. In the construction we use the property of abstract network design problems of admitting a min operator. We first prove the following theorem:

Theorem 6.1. *Let P be an Abstract Network Design problem with load function ρ. Suppose that P admits a min operator and that its baseline algorithm* BaseAlg *has competitive ratio β. Furthermore, suppose that there exists an algorithm Alg^T that is α-competitive on instances defined on HST metrics. Then, there exists a randomized algorithm* CombineAlg *that, on every instance, has expected competitive ratio*

$$O(\alpha \cdot \min\{\log k \cdot \log(\beta k \lambda_\rho), \log^2 k \cdot \log(\beta k)\}),$$

where $\lambda_\rho = \min\{\lceil \max_S \rho(S) / \min_{S \neq \emptyset} \rho(S) \rceil, r\}$.

Since the dependency on the competitive ratio of the baseline algorithm decreases exponentially, this idea can be applied repeatedly, to completely remove the dependency on the original baseline algorithm as well. This yields Theorem 1.7.

We begin by describing the high-level intuition behind CombineAlg and its analysis. Typically, tree embeddings are used to give a reduction from problems on general graphs to trees. This reduction consists of three steps: (1) compute a tree embedding T of the input graph G; (2) solve the problem on T; (3) translate the solution on T back to G. The analysis relies on the fact that the tree embedding is non-contractive to argue that the cost of the translated solution is at most the cost of the tree solution. Then, it bounds the cost of the optimal solution in T in terms of the optimal solution in G. This is done by considering the translation of the optimal solution in G into T and upper bounding the blow up in cost by the distortion of the embedding.

Our approach uses the combining scheme of the min operator property to combine the usual tree-embedding-based algorithm (using the embedding of Theorem 4.1) and the baseline algorithm.

At a high level, this allows us to fall back on the baseline algorithm in the case that the optimal solution in T becomes very expensive due to some edges of G being distorted badly. In the analysis, we first show (Lemma 6.2) that the resulting algorithm has cost at most $O(\alpha)\min\{\mathrm{OPT}(T), \beta\,\mathrm{OPT}\}$, where $\mathrm{OPT}(T)$ is the cost of the optimal solution in T and OPT is the cost of the optimal solution in G. We then bound $\min\{\mathrm{OPT}(T), \beta\,\mathrm{OPT}\}$ by considering the translation of the optimal solution in G into T using a more refined analysis to bound the overhead of the translation.

At a high level, the embedding of Theorem 4.1 uses a collection of $O(\log \Phi)$ probabilistic partitions, one per scale. Each partition contributes a $O(\log k)$ factor to the distortion, resulting in an overall distortion of $O(\log k \log \Phi)$. The key idea is that to bound $\min\{\mathrm{OPT}(T), \beta\,\mathrm{OPT}\}$, it suffices to only consider much fewer than $O(\log \Phi)$ scales. It is straightforward to see that we can ignore scales above $\beta\,\mathrm{OPT}$. Arguing that scales much smaller than OPT do not contribute much to the cost of the translated solution is more difficult. If there exists a near-optimal solution in G that uses at most $\mathrm{poly}(k)$ edges, then we can ignore scales that are smaller than $\mathrm{OPT}/\mathrm{poly}(k)$ and so only $\log(\beta k)$ scales are relevant. However, in general it is unclear that such a bound is possible. Instead, we use a more subtle argument (Lemma 6.5) based on a notion of sparsity that we call "tree sparsity" (Definition 6.3). We apply this argument in two different ways: one based on the parameter λ_ρ and the fact that T has $O(k)$ edges; the other based on the fact that one can use offline tree embeddings to obtain a $O(\log k)$-approximate solution in G that uses at most $O(k)$ edges. The minimum of the resulting bounds yields the bound in Theorem 6.1.

Description of CombineAlg.

Let $\mathsf{Alg}^{\mathcal{T}}$ be an algorithm that is α-competitive on HST instances and BaseAlg to be a baseline algorithm that is β-competitive on arbitrary instances. Let $\mathsf{Embed}^{\mathcal{T}}$ be the online embedding algorithm of Theorem 4.1 and Alg be the algorithm of Theorem 3.4 that uses $\mathsf{Embed}^{\mathcal{T}}$ as its online tree embedding algorithm. The algorithm CombineAlg is the combination of $\mathsf{Alg}^{\mathcal{T}}$ and Alg.

6.1 Analysis

For a fixed choice of μ-HST metric (T, d_T) used by CombineAlg, let $\mathrm{ALG}(T)$ denote the cost of CombineAlg. Observe that for the problem on T, the subgraphs of any feasible solution must be a subgraph of T.

Lemma 6.2. $\mathrm{ALG}(T) \le O(\alpha)\min\{\mathrm{OPT}(T), \beta\,\mathrm{OPT}\}$. *When $\mathsf{Alg}^{\mathcal{T}}$ and/or BaseAlg are randomized algorithms, then we have $E[\mathrm{ALG}(T)] \le O(\alpha)\min\{\mathrm{OPT}(T), \beta\,\mathrm{OPT}\}$ where the expectation is over the internal randomness of $\mathsf{Alg}^{\mathcal{T}}$ and/or BaseAlg (but not over the random choice of T).*

Proof. The lemma follows by applying Claim 3.5 together with Defintion 1.6. □

We now bound $\mathrm{OPT}(T)$ in terms of any feasible solution $\mathcal{S}$ in G. Let $\mathcal{R}_i = (R_i, C_i)$ be its i-th response. Consider the solution $\mathcal{S}'$ in T obtained by translating $\mathcal{S}$ into T using the extension function $F := F_r$ as follows. For an edge $e = (u, v) \in G$, let $I_e = \{i : e \in R_i\}$ and $P_T(e)$ denote the path in T between $F(u)$ and $F(v)$. The i-th response of $\mathcal{S}'$ is (R'_i, C_i) where $R'_i = \bigcup_{e \in R_i} P_T(e)$. Since any terminal pair that is connected in R_i is also connected in R'_i, the solution $\mathcal{S}'$ is a feasible solution on T. Thus, $\mathrm{OPT}(T) \le \mathrm{cost}(\mathcal{S}')$. Since each tree edge $e' \in T$ is used by $\mathcal{S}'$ in the time steps i such that R_i contains an edge $e \in G$ with $e' \in P_T(e)$, we have

$$\mathrm{OPT}(T) \leq \sum_{e' \in T} d_T(e') \rho \left(\bigcup_{e \in G : e' \in P_T(e)} I_e \right). \tag{1}$$

While one can use subadditivity to upper bound the RHS of the inequality by $\sum_{e \in G} d_T(e)\rho(I_e)$, we prove a more refined bound based on the following notion of sparsity.

Definition 6.3 (Tree sparsity)**.** *Consider a solution $\mathcal{S}$ in G. For a HST embedding T of G, let $\widehat{E}_T$ be the smallest edge subset of G such that $\sum_{e' \in T} d_T(e')\rho\left(\bigcup_{e \in G : e' \in P_T(e)} I_e\right) \leq \sum_{e \in \widehat{E}_T} d_T(e)\rho(I_e)$. The* tree sparsity *of $\mathcal{S}$ is the maximum size of $\widehat{E}_T$ over non-expansive HST embeddings of G.*

Intuitively, if a solution $\mathcal{S}$ has small tree sparsity, then for any HST embedding T, the cost of its translation into T can be charged to a small subset of edges in G. Note that subadditivity of ρ immediately implies that $|\cup_i R_i|$ is an upper bound on the tree sparsity of $\mathcal{S}$.

We will need the following technical claim in the proof of the upcoming lemma.

Claim 6.4. *Let (V, d) be a metric space and $t_1, \dots, t_k$ be a sequence of k terminals. Consider the μ-HST embedding T of Theorem 4.1. Then, for any $e = (u, v)$ with $u, v \in V$ and $\gamma, \sigma, \delta > 0$ such that $\gamma > \delta$, the random variable*

$$\widehat{d}_T(e) = \begin{cases} \gamma & \text{for } \sigma d_T(e) \geq \gamma \\ \sigma d_T(e) & \text{for } \delta \leq \sigma d_T(e) < \gamma \\ 0 & \text{for } \sigma d_T(e) < \delta \end{cases}$$

has expectation (over the choice of T) $E_T[\widehat{d}_T(e)] \leq O(\log k \cdot \log \frac{\gamma}{\delta}) \cdot \sigma d(e)$.

Proof. For an event $\mathcal{E}$, let $1\{\mathcal{E}\}$ be its indicator variable. Since T is a μ-HST, we have

$$\widehat{d}_T(e) \leq \gamma \cdot 1\{d_T(e) \geq \gamma/\sigma\} + O(1) \sum_{j : \mu^j \in [\delta, \gamma]} \mu^j \cdot 1\{d_T(e) \geq \mu^j/\sigma\}.$$

and so

$$E_T[\widehat{d}_T(e)] \leq \gamma \cdot \Pr_T[d_T(e) \geq \gamma/\sigma] + O(1) \sum_{j : \mu^j \in [\delta, \gamma]} \mu^j \cdot \Pr_T[d_T(e) \geq \mu^j/\sigma].$$

By Theorem 4.1, $\Pr_T[d_T(e) \geq \mu^j/\sigma] \leq O(\log k)\frac{d(e)}{\mu^j/\sigma}$ for each j and $\Pr_T[d_T(e) \geq \gamma/\sigma] \leq O(\log k)\frac{d(e)}{\gamma/\sigma}$. As there are at most $O(\log \frac{\gamma}{\delta})$ terms in the sum, we have $E_T[\widehat{d}_T(e)] \leq O(\log k \cdot \log \frac{\gamma}{\delta}) \cdot \sigma d(e)$, as desired. ☐

Lemma 6.5. *Let $\mathcal{S}$ be a solution in G with tree sparsity η. Then, $E_T[\min\{\mathrm{OPT}(T), \beta\,\mathrm{OPT}\}] \leq O(\log k \log(\beta\eta))\,\mathrm{cost}(\mathcal{S})$.*

Proof. Fix a HST embedding T. Let $\phi_T(e) = d_T(e)\rho(I_e)$. Since $\mathcal{S}$ has tree sparsity η, there exists $\widehat{E}_T \subseteq G$ of size at most η with $\sum_{e' \in T} d_T(e')\rho\left(\bigcup_{e \in G : e' \in P_T(e)} I_e\right) \leq \sum_{e \in \widehat{E}_T} \phi_T(e)$. Using Inequality (1), we get

$$\mathrm{OPT}(T) \leq \sum_{e \in \widehat{E}_T} \phi_T(e) \leq \sum_{e \in \widehat{E}_T : \phi_T(e) \geq \frac{\mathrm{OPT}}{\eta}} \phi_T(e) + \mathrm{OPT} \leq \sum_{e \in G : \phi_T(e) \geq \frac{\mathrm{OPT}}{\eta}} \phi_T(e) + \mathrm{OPT},$$

where the last inequality uses the fact that $\widehat{E}_T$ is a subset of edges in G. Thus, it suffices to bound $E_T[\min\{\sum_{e\in G:\phi_T(e)\geq \frac{\text{OPT}}{\eta}} \phi_T(e), \beta\,\text{OPT}\}]$. Now, observe that

$$\min\left\{\sum_{e\in G:\phi_T(e)\geq \frac{\text{OPT}}{\eta}} \phi_T(e), \beta\,\text{OPT}\right\} \leq \sum_{e\in G:\phi_T(e)\geq \frac{\text{OPT}}{\eta}} \min\{\phi_T(e), \beta\,\text{OPT}\}$$

$$= \sum_{e\in G} 1\{\phi_T(e) \geq \frac{\text{OPT}}{\eta}\} \min\{\phi_T(e), \beta\,\text{OPT}\},$$

where $1\{\phi_T(e) \geq \frac{\text{OPT}}{\eta}\}$ is the indicator variable for the event $\phi_T(e) \geq \frac{\text{OPT}}{\eta}$. Consider the random variable $\widehat{d}_T(e)$ where

$$\widehat{d}_T(e) = \begin{cases} \beta\,\text{OPT} & \text{for } d_T(e)\rho(I_e) \geq \beta\,\text{OPT} \\ d_T(e)\rho(I_e) & \text{for } \frac{\text{OPT}}{\eta} \leq d_T(e)\rho(I_e) < \beta\,\text{OPT} \\ 0 & \text{for } d_T(e)\rho(I_e) < \frac{\text{OPT}}{\eta} \end{cases}$$

We have that

$$\widehat{d}_T(e) = 1\{\phi_T(e) \geq \frac{\text{OPT}}{\eta}\} \min\{\phi_T(e), \beta\,\text{OPT}\}.$$

So,

$$E_T\left[\min\left\{\sum_{e\in G:\phi_T(e)\geq \frac{\text{OPT}}{\eta}} \phi_T(e), \beta\,\text{OPT}\right\}\right] \leq \sum_{e\in G} E_T[\widehat{d}_T(e)] \leq O(\log k \cdot \log(\beta\eta)) \sum_{e\in G} d(e)\rho(I_e),$$

where the last inequality follows from applying Proposition 6.4 to each $e \in G$ with $\gamma = \beta\,\text{OPT}, \delta = \text{OPT}/\eta, \sigma = \rho(I_e)$. Since $\text{cost}(\mathcal{S}) = \sum_{e\in G} d(e)\rho(I_e)$, this completes the proof of the lemma. □

With Lemma 6.5 in hand, to complete the proof of Theorem 6.1, it remains to show that: (1) any optimal solution has tree sparsity $O(k\lambda_\rho)$ (Lemma 6.7); (2) there exists an $O(\log k)$-approximate solution with tree sparsity $k-1$ (Lemma 6.10). We first show (1).

Proposition 6.6. *For any collection $\mathcal{A}$ of nonempty subsets of $\{1,\ldots,r\}$, there exists a subcollection $\widehat{\mathcal{A}} \subseteq \mathcal{A}$ of size at most λ_ρ such that $\rho\left(\bigcup_{A\in\mathcal{A}} A\right) \leq \sum_{A\in\widehat{\mathcal{A}}} \rho(A)$.*

Proof. Case 1: $\lambda_\rho = \lceil \rho_{\max}/\rho_{\min}\rceil$. If $|\mathcal{A}| \leq \lambda_\rho$, the proposition follows trivially by taking $\widehat{\mathcal{A}} = \mathcal{A}$ and using the subadditivity of ρ. Suppose that $|\mathcal{A}| > \lambda_\rho$, let $\widehat{\mathcal{A}}$ be any subcollection of $\mathcal{A}$ of size λ_ρ. Observe that $\sum_{A\in\widehat{\mathcal{A}}} \rho(A) \geq \lambda_\rho \min_{S\neq\emptyset} \rho(S)\cdot \geq \max_S \rho(S) \geq \rho\left(\bigcup_{A\in\mathcal{A}} A\right)$. Case 2: $\lambda_\rho = r$. Construct $\widehat{\mathcal{A}}$ by greedily choosing sets from $\mathcal{A}$, iteratively choosing a set which adds at least one new element of $\{1,\ldots,r\}$, until the union of sets of $\widehat{\mathcal{A}}$ equals that of $\mathcal{A}$. Then $|\widehat{\mathcal{A}}| \leq r$ and the result follows from subadditivity of ρ. □

Lemma 6.7. *Any optimal solution has tree sparsity $O(k\lambda_\rho)$.*

Proof. Let $\mathcal{S}$ be an optimal solution and T be a HST embedding. For each $e' \in T$, let $\widehat{E}_T(e')$ be a subset of at most λ_ρ edges $e \in G$ with $e' \in P_T(e)$ such that $\rho\left(\bigcup_{e \in G: e' \in P_T(e)} I_e\right) \leq \sum_{e \in \widehat{E}_T(e')} \rho(I_e)$ (such a subset exists by Proposition 6.6). Let $\widehat{E}_T = \bigcup_{e' \in T} \widehat{E}_T(e')$; note that $|\widehat{E}_T| \leq O(k\lambda_\rho)$ since T has at most $O(k)$ edges (by Theorem 4.1). So, we have

$$\begin{aligned}\sum_{e' \in T} d_T(e') \rho\left(\bigcup_{e \in G: e' \in P_T(e)} I_e\right) &\leq \sum_{e' \in T} d_T(e') \sum_{e \in \widehat{E}_T(e')} \rho(I_e) \\ &= \sum_{e \in \widehat{E}_T} \left(\sum_{e' \in T: e \in \widehat{E}_T(e')} d_T(e')\right) \rho(I_e) \\ &\leq \sum_{e \in \widehat{E}_T} d_T(e) \rho(I_e)\end{aligned}$$

where the last inequality follows from the fact that the set of tree edges e' with $e \in \widehat{E}_T(e')$ is a subset of the path in T between the endpoints of e, and the total length of the path in T is exactly $d_T(e)$. □

Finally, we prove (2). We will need the following observation from Gupta, Nagarajan, and Ravi [GNR10, Theorem 7], based on the probabilistic embedding of [FRT04, Bar04], and a claim on Steiner points removal in HST trees (see e.g. Theorem 5.1 in [KRS01] or the construction of the extension H_i in the proof of Claim C.7 in Appendix C).

Theorem 6.8. *Let (M, d_M) be a metric space with a designated subset $W \subseteq M$. Then there is a distribution $\mathcal{T}$ of HSTs with leaves M such that for every $T \in \mathcal{T}$, we have $d_T(x, y) \geq d_M(x, y)$ for every $x, y \in W$ and $E_{T \sim \mathcal{T}}[d_T(u, v)] \leq O(\log |W|) d_M(u, v)$ for every $u, v \in M$.*

Claim 6.9. *Let (T, d_T) be a HST metric and Z be a subset of its vertices. Then there exists an embedding g of T into a tree metric $(T', d_{T'})$ whose vertex set is exactly Z such that $d_{T'}(g(u), g(v)) \leq d_T(u, v)$ for every $u, v \in T$ and $d_T(x, y) \leq 4 d_{T'}(g(x), g(y))$ for every $x, y \in Z$.*

Lemma 6.10. *There exists a $O(\log k)$-approximate solution with tree sparsity $k - 1$.*

Proof. Theorem 6.8 implies that there exists a HST embedding T such that $\mathrm{OPT}(T) \leq O(\log k)\,\mathrm{OPT}$ and $d_T(u, v) \geq d(u, v)$ for every pair of terminals $u, v \in \mathcal{Z}$. Let T' and g be the tree metric and the embedding guaranteed by applying Claim 6.9 to $\mathcal{Z}$ and the subtree of T induced by $\mathcal{Z}$. Let $\mathrm{OPT}(T')$ be the cost of the optimal solution for the instance on T' induced by g. We now argue that any feasible solution on T can be transformed into a feasible solution on T' of less or equal cost. Let $\mathcal{S} = ((R_1, C_1), \ldots, (R_r, C_r))$ be a feasible solution on T and $\mathcal{S}' = ((R'_1, C'_1), \ldots, (R'_r, C'_r))$ with $R'_i = g(R_i)$ and $C'_i = g(C_i)$. The solution $\mathcal{S}'$ is feasible for the instance on T' induced by g, and $\mathrm{cost}(\mathcal{S}') \leq \mathrm{cost}(\mathcal{S})$. Thus, $\mathrm{OPT}(T') \leq \mathrm{OPT}(T) \leq O(\log k)\,\mathrm{OPT}$.

Let $\mathcal{S}^*$ be the optimal solution for T' and $R^*_1, \ldots, R^*_r$ be the subgraphs it uses. Since the vertices of T' is a subset of G (in particular, it is exactly the set of terminals $\mathcal{Z}$), $\mathcal{S}^*$ is also a feasible solution in G. Moreover, since $d(u, v) \leq d_T(f(u), f(v)) \leq 4 d_{T'}(g(f(u)), g(f(v)))$ for every $u, v \in \mathcal{Z}$, the cost of $\mathcal{S}^*$ in G is at most $4\,\mathrm{OPT}(T') \leq O(\log k)\,\mathrm{OPT}$. Finally, the tree sparsity of $\mathcal{S}^*$ is at most $|\cup_i R'_i| \leq k - 1$ since T' is a tree with k vertices. □

Combining Lemma 6.5 with Lemmas 6.7 and 6.10 gives us Theorem 6.1.

References

[AA97] Baruch Awerbuch and Yossi Azar. Buy-at-bulk network design. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 542–547, 1997.

[AAA+06] Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. A general approach to online network optimization problems. *ACM Trans. Algorithms*, 2(4):640–660, 2006.

[AAB04] Baruch Awerbuch, Yossi Azar, and Yair Bartal. On-line generalized steiner problem. *Theor. Comput. Sci.*, 324(2-3):313–324, 2004.

[ABM93] Yossi Azar, Andrei Z. Broder, and Mark S. Manasse. On-line choice of on-line algorithms. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, pages 432–440, Philadelphia, PA, USA, 1993. Society for Industrial and Applied Mathematics.

[ABN06] Ittai Abraham, Yair Bartal, and Ofer Neiman. Advances in metric embedding theory. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 271–286, 2006.

[AC07] Noga Alon and Michael R. Capalbo. Finding disjoint paths in expanders deterministically and online. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 518–524, 2007.

[And04] Matthew Andrews. Hardness of buy-at-bulk network design. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 115–124, 2004.

[Bar96] Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996.

[Bar04] Yair Bartal. Graph decomposition lemmas and their role in metric embedding methods. In *Algorithms - ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings*, pages 89–97, 2004.

[BBBT97] Yair Bartal, Avrim Blum, Carl Burch, and Andrew Tomkins. A polylog(n)-competitive algorithm for metrical task systems. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 711–719, 1997.

[BBMN15] Nikhil Bansal, Niv Buchbinder, Aleksander Madry, and Joseph Naor. A polylogarithmic-competitive algorithm for the k-server problem. *J. ACM*, 62(5):40:1–40:49, 2015.

[BC97] Piotr Berman and Chris Coulston. On-line algorithms for steiner tree problems (extended abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 344–353, 1997.

[BCI01] Yair Bartal, Moses Charikar, and Piotr Indyk. On page migration and other relaxed task systems. *Theor. Comput. Sci.*, 268(1):43–66, 2001.

[BCL+18] Sebastien Bubeck, Michael B. Cohen, Yin Tat Lee, James R. Lee, and Aleksander Madry. K-server via multiscale entropic regularization. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 3–16, New York, NY, USA, 2018. ACM.

[BCR01] Yair Bartal, Moses Charikar, and Danny Raz. Approximating min-sum k-clustering in metric spaces. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 11–20, 2001.

[ER17] Matthias Englert and Harald Räcke. Reordering buffers with logarithmic diameter dependency for trees. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, January 16-19*, pages 1224–1234, 2017.

[ERW10] Matthias Englert, Harald Räcke, and Matthias Westermann. Reordering buffers for general metric spaces. *Theory of Computing*, 6(1):27–46, 2010.

[FRT04] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.

[GHR06] Anupam Gupta, Mohammad Taghi Hajiaghayi, and Harald Räcke. Oblivious network design. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006*, pages 970–979, 2006.

[GI06] Fabrizio Grandoni and Giuseppe F. Italiano. Improved approximation for single-sink buy-at-bulk. In *Algorithms and Computation, 17th International Symposium, ISAAC 2006, Kolkata, India, December 18-20, 2006, Proceedings*, pages 111–120, 2006.

[GKK+01] Naveen Garg, Rohit Khandekar, Goran Konjevod, R. Ravi, F. Sibel Salman, and Amitabh Sinha II. On the integrality gap of a natural formulation of the single-sink buy-at-bulk network design problem. In *Integer Programming and Combinatorial Optimization, 8th International IPCO Conference, Utrecht, The Netherlands, June 13-15, 2001, Proceedings*, pages 170–184, 2001.

[GKR98] Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, 25-27 January 1998, San Francisco, California.*, pages 253–259, 1998.

[GKR04] Anupam Gupta, Amit Kumar, and Rajeev Rastogi. Traveling with a pez dispenser (or, routing issues in MPLS). *SIAM J. Comput.*, 34(2):453–474, 2004.

[GMM09] Sudipto Guha, Adam Meyerson, and Kamesh Munagala. A constant factor approximation for the single sink edge installation problem. *SIAM J. Comput.*, 38(6):2426–2442, 2009.

[GNR10] Anupam Gupta, Viswanath Nagarajan, and R. Ravi. An improved approximation algorithm for requirement cut. *Oper. Res. Lett.*, 38(4):322–325, 2010.

[GP12] Ashish Goel and Ian Post. One tree suffices: A simultaneous o(1)-approximation for single-sink buy-at-bulk. *Theory of Computing*, 8(1):351–368, 2012.

[GR10] Fabrizio Grandoni and Thomas Rothvoß. Network design via core detouring for problems without a core. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, pages 490–502, 2010.

[GRTU17] Anupam Gupta, R. Ravi, Kunal Talwar, and Seeun William Umboh. LAST but not least: Online spanners for buy-at-bulk. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 589–599, 2017.

[GW95] Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.

[IMSZ10] Piotr Indyk, Avner Magen, Anastasios Sidiropoulos, and Anastasios Zouzias. Online embeddings. In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 246–259, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[Ind01] P. Indyk. Algorithmic applications of low-distortion geometric embeddings. In *Proceedings of the 42Nd IEEE Symposium on Foundations of Computer Science*, FOCS '01, pages 10–, Washington, DC, USA, 2001. IEEE Computer Society.

[IW91] Makoto Imase and Bernard M. Waxman. Dynamic steiner tree problem. *SIAM J. Discrete Math.*, 4(3):369–384, 1991.

[JR09] Raja Jothi and Balaji Raghavachari. Improved approximation algorithms for the single-sink buy-at-bulk network design problems. *J. Discrete Algorithms*, 7(2):249–255, 2009.

[KPR93] Philip Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, STOC '93, pages 682–690, New York, NY, USA, 1993. ACM.

[KRS01] Goran Konjevod, R. Ravi, and F. Sibel Salman. On approximating planar metrics by tree metrics. *Inf. Process. Lett.*, 80(4):213–219, 2001.

[KT02] Jon M. Kleinberg and Éva Tardos. Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. *J. ACM*, 49(5):616–639, 2002.

[Lee18] James R. Lee. Fusible hsts and the randomized k-server conjecture. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 438–449, 2018.

[LPS88] Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[LS91] Nathan Linial and Michael Saks. Decomposing graphs into regions of small diameter. In *SODA '91: Proceedings of the second annual ACM-SIAM symposium on Discrete algorithms*, pages 320–330, Philadelphia, PA, USA, 1991. Society for Industrial and Applied Mathematics.

[NPS11] Joseph Naor, Debmalya Panigrahi, and Mohit Singh. Online node-weighted steiner tree and related problems. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 210–219, 2011.

[RR98] Yuri Rabinovich and Ran Raz. Lower bounds on the distortion of embedding finite metric spaces in graphs. *Discrete & Computational Geometry*, 19(1):79–94, 1998.

[Tal02] Kunal Talwar. The single-sink buy-at-bulk LP has constant integrality gap. In *Integer Programming and Combinatorial Optimization, 9th International IPCO Conference, Cambridge, MA, USA, May 27-29, 2002, Proceedings*, pages 475–486, 2002.

[Umb15] Seeun Umboh. Online network design algorithms via hierarchical decompositions. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1373–1387, 2015.

A Metric Completion of an Input Graph

Claim A.1. *Let P be an Abstract Network Design problem and let G be its input graph. Then, w.l.o.g. G is a complete graph with edge length satisfying triangle inequality.*

Proof. We construct a complete graph $\hat{G}$ as follows: remove all the edges in G that form triangles yet do not satisfy the triangle inequality, and complete the graph by adding edges $e = (u, v)$ with $d(e)$ being the length of a shortest path in G between u and v.

Let $\hat{\mathcal{S}}_i = ((\hat{R}_1, \hat{C}_1), \ldots, (\hat{R}_i, \hat{C}_i))$ denote a feasible solution to P with the input graph being $\hat{G}$. We show that there is a feasible solution $\mathcal{S}_i = ((R_1, C_1) \ldots, (R_i, C_i))$ for P with the input graph being G, such that $\text{cost}(\mathcal{S}_i) \leq \text{cost}(\hat{\mathcal{S}}_i)$. For each $\hat{R}_j \in \hat{\mathcal{S}}_i$, for all $e = (u, v) \in \hat{R}_j$ that either has been added to $\hat{R}_j$ or the weight of which has been updated in $\hat{R}_j$ let $p_G(e)$ denote a shortest path between u and v in G. We add the edges of the path $p_G(e)$ to the response R_j. All the other edges in $\hat{R}_j$ are added to R_j as well. The connectivity lists C_j are defined to be exactly the lists $\hat{C}_j$.

By the construction, the solution $\mathcal{S}_i$ is feasible. In addition, if $e' \in \hat{\mathcal{R}}_j$ is such that $e' \notin R_j$ or such that $d_G(e') \neq d_{\hat{G}}(e')$, then by the construction of $\hat{G}$, $d_{\hat{G}}(e')$ equals to the length $p_G(e') \subseteq R_j$. Thus, by the subadditivity of the load function we conclude that $\text{cost}(\mathcal{S}_i) \leq \text{cost}(\hat{\mathcal{S}}_i)$.

On the other hand, let $\mathcal{S}_i^*$ be an optimal solution on the input graph G. Note that since its response subgraphs R_j^* do not use edges that violate triangle inequality, the solution $\mathcal{S}_i^*$ is also a feasible solution on $\hat{G}$, of the same cost. This implies that the costs of an optimal solution on G and on $\hat{G}$ are equal. This completes the proof. □

B Combining Online Algorithms

Let P be an online problem from an Abstract Network Design model. For a deterministic online algorithm A and for a sequence of $i \geq 1$ requests $(Z_1, \ldots, Z_i)$, let $\mathcal{S}_i^A = (\mathcal{R}_1^A, \ldots, \mathcal{R}_i^A)$ denote the feasible solution of A.

Definition B.1. *[Restatement of Definition 1.6] We say that an online problem P admits a min operator with factor $\eta \geq 1$ if*

1. *There is a β-competitive online algorithm for P, for some β;*
2. *For any given deterministic online algorithms A and B, there is an online (possibly randomized) algorithm C, such that for any $(Z_1, \ldots, Z_r)$ requests to P, $\text{cost}(\mathcal{S}_r^C) \leq \eta \cdot \min\{\text{cost}(\mathcal{S}_r^A), \text{cost}(\mathcal{S}_r^B)\}$.*

Definition B.2 (Sequencing property)**.** *We say that a problem P* satisfies the sequencing property with parameters (a, b), $a, b \geq 1$, *if for any (possibly online) algorithms $\hat{A}$ and $\hat{B}$, for any $(Z_1, \ldots, Z_i)$ requests and for any sequence of new $t - i$ requests $(Z_{i+1}, \ldots, Z_t)$, for any $t > i$, there exists a sequence of responses $((R'_{i+1}, C'_{i+1}), \ldots, (R'_t, C'_t))$ such that*

1. *$\mathcal{F}_t(C_1^{\hat{A}}, \ldots, C_i^{\hat{A}}, C'_{i+1}, \ldots, C'_t) = 1$;*
2. *$\text{cost}(\mathcal{R}_1^{\hat{A}}, \ldots, \mathcal{R}_i^{\hat{A}}, \mathcal{R}'_{i+1}, \ldots, \mathcal{R}'_t) \leq a \cdot \text{cost}(\mathcal{R}_1^{\hat{A}}, \ldots, \mathcal{R}_i^{\hat{A}}) + b \cdot \text{cost}(\mathcal{R}_1^{\hat{B}}, \ldots, \mathcal{R}_t^{\hat{B}})$.*

We next show that the sequencing property is enough to ensure the problem admits min operator.

Lemma B.3. *If an Abstract Network Design problem P satisfies the sequencing property with parameters (a, b), then P admits min operator with factor $\eta = O(a^{3.5} \cdot b)$, and competitive ratio $\beta = a^{\Theta(r)} \cdot b$.*

Proof. To show the existence of a competitive algorithm for P, let A_{OPT_i} denote the algorithm that receives the sequence of requests $(Z_1, \ldots, Z_i)$, all at once, and outputs the optimal offline responses to these requests, denoted by $((R_1^{\text{OPT}_i}, C_1^{\text{OPT}_i}), \ldots, (R_i^{\text{OPT}_i}, C_i^{\text{OPT}_i}))$. The competitive algorithm proceeds as follows: for the first request Z_1, output $(R_1^{\text{OPT}_i}, C_1^{\text{OPT}_i})$; for the request Z_2, define $\hat{A} := A_{\text{OPT}_1}$ and $\hat{B} := A_{\text{OPT}_2}$. By the definition of the sequencing property, there exists a response (R'_2, C'_2), such that $\mathcal{F}_2(C_1^{\text{OPT}_1}, C'_2) = 1$, and $\text{cost}(\mathcal{R}_1^{\text{OPT}_1}, \mathcal{R}'_2) \leq a \cdot \text{cost}(\mathcal{R}_1^{\text{OPT}_1}) + b \cdot \text{cost}(\mathcal{R}_1^{\text{OPT}_2}, \mathcal{R}_2^{\text{OPT}_2}) \leq a \cdot \text{OPT}_1 + b \cdot \text{OPT}_2 \leq (a + b)\, \text{OPT}_2$. For the request Z_3, let $\hat{A}$ be the algorithm that outputs $((R_1^{\text{OPT}_1}, C_1^{\text{OPT}_1}), (R'_2, C'_2))$ on the first two requests and $\hat{B} := A_{\text{OPT}_3}$, conclude similarly the existence of a response (R'_3, C'_3) such that $\text{cost}(\hat{A}) \leq (a(a + b) + b)\, \text{OPT}_3$. Continue in the same way, to obtain a $(a^{\Theta(r)} \cdot b)$-competitive algorithm for P.

Next we prove the second property of the min operator. Let A and B be any online algorithms for P. For a sequence of requests $(Z_1, \ldots, Z_t)$ let $\text{cost}_A(t)$ (let $\text{cost}_B(t)$) denote the cost of the solution of the algorithm A (the algorithm B) on the sequence $(Z_1, \ldots, Z_t)$. Let M denote $\min\{A, B\}$ (i.e., the algorithm of smaller cost).

The algorithm C proceeds in phases. Let $t_1 > 1$ be the maximal index such that for the sequence of requests $(Z_1, \ldots, Z_{t_1})$ it holds that $\text{cost}_A(t_1) \leq \gamma \cdot \text{cost}_B(t_1)$ for some $\gamma \geq 1$ that will be chosen later, and let $t'_2 = t_1 + 1$. Note that $\text{cost}_A(t'_2) > \gamma \cdot \text{cost}_B(t'_2)$ by the definition of t_1. We say that the first phase occurs at the timesteps $[1, t_1]$. The responses of C during the first phase are the responses of A. Let C_1 denote the algorithm C in the first phase.

For $i > 1$ the phase i is defined by the following: the phase occurs at the timesteps $[t'_i, t_i]$, where $t_i > t'_i$ is defined as the maximal index for which $\text{cost}_B(t_i) \leq \gamma \cdot \text{cost}_A(t_i)$ if i is even, and $t_i > t'_i$ is the maximal index for which $\text{cost}_A(t_i) \leq \gamma \cdot \text{cost}_B(t_i)$ if i is odd. Let $\hat{A}_i = C_{i-1}$; let $\hat{B}_i = B$ if i is even and $\hat{B}_i = A$ if i is odd. Since P satisfies sequencing property it holds that for $\hat{A}$ and $\hat{B}$, for any sequence of new requests $(Z_{t'_i}, \ldots, Z_{t_j})$, for any $t_j \geq t'_i$, there are responses $((R'_{t'_i}, C'_{t'_i}), \ldots, (R'_{t_j}, C'_{t_j}))$ such that $\text{cost}(\mathcal{R}_1^{\hat{A}}, \ldots, \mathcal{R}_{t_{i-1}}^{\hat{A}}, \mathcal{R}'_{t'_i}, \ldots, \mathcal{R}'_{t_j}) \leq a \cdot \text{cost}_{\hat{A}}(t_{i-1}) + b \cdot \text{cost}_{\hat{B}}(t_j)$. We define algorithm C_i to be the algorithm that outputs the responses of C_{i-1} followed by the responses obtained from the sequencing algorithm $\hat{A}$ with $\hat{B}$, i.e., the responses $\{(R'_{t'_l}, C'_{t'_l})\}_{l \geq 1}$.

We prove the following claim:

Claim B.4. *Let $\zeta = 4a^{1.5}(a^2 + a + 1)b$ and $\gamma = 2a^{1.5}$. For all $i \geq 2$, for all $t \in [t'_i, t_i]$ it holds that $\text{cost}_{C_i}(t) \leq \zeta \cdot \text{cost}_M(t)$.*

Proof. The proof is by induction on the phase $i \geq 3$. Let $i \geq 3$ and $t \in [t'_i, t_i]$. By the definition of the algorithm C_i and by the sequencing property, we have $\text{cost}_{C_i}(t) \leq a \cdot \text{cost}_{C_{i-1}}(t_{i-1}) + b \cdot \text{cost}_{\hat{B}_i}(t) \leq a \cdot \text{cost}_{C_{i-1}}(t_{i-1}) + b \cdot \gamma \cdot \text{cost}_M(t)$, where the second inequality holds by definition of $\hat{B}_i$. Applying this rule on $\text{cost}_{C_{i-1}}(t_{i-1})$ for two more steps, we obtain $\text{cost}_{C_i}(t) \leq a^3 \cdot \text{cost}_{C_{i-3}}(t_{i-3}) + a^2(a+1)b\gamma \cdot \text{cost}_M(t)$. By induction assumption it holds that $\text{cost}_{C_{i-3}}(t_{i-3}) \leq \zeta \cdot \text{cost}_M(t_{i-3})$. Bellow we show that for all $t > t'_i$, $\text{cost}_M(t) \geq \gamma^2 \cdot \text{cost}_M(t_{i-3})$. Thus, we conclude: $\text{cost}_{C_i}(t) \leq (a^3\zeta/\gamma^2 + a^2(a+1)b\gamma) \cdot \text{cost}_M(t) \leq \zeta \, \text{cost}_M(t)$ for chosen values of ζ and γ.

It remains to show that for all $t > t'_i$, $\text{cost}_M(t) \geq \gamma^2 \cdot \text{cost}_M(t_{i-3})$. By the monotonicity of the load function, it is enough to prove that $\text{cost}_M(t'_i) \geq \gamma^2 \, \text{cost}_M(t'_{i-2})$. Assume w.l.o.g. that at the phase i it holds that $\text{cost}_B(t) \leq \gamma \, \text{cost}_A(t)$ for all timesteps t in the phase (i.e., assume that i is an even phase). Then, by the definition of t'_i, it holds that $\text{cost}_A(t'_i) > \gamma \, \text{cost}_B(t'_i)$, i.e., $M(t'_i) = B$. Similarly, by the definition of t'_{i-2}, we conclude that $M(t'_{i-2}) = B$. Since $\text{cost}_M(t) = \text{cost}_B(t) \geq \text{cost}_B(t'_{i-1}) > \gamma \, \text{cost}_A(t'_{i-1}) \geq \gamma \, \text{cost}_A(t'_{i-2}) > \gamma^2 \, \text{cost}_B(t'_{i-2}) \geq \gamma^2 \, \text{cost}_B(t_{i-3})$, which completes the proof. □

□

This completes the proof of the lemma.

Corollary B.5. *If the feasibility requirement of an Abstract Network Design problem P is memoryless then P admits a min operator.*

Proof. By Lemma B.3, it is enough to show that P satisfies the sequencing property. Assume we are given two deterministic online algorithms A and B. For any $t > i \geq 1$, for a sequence of new requests $(Z_{i+1}, \ldots, Z_t)$ let $((R^B_{i+1}, C^B_{i+1}), \ldots, (R^B_t, C^B_t))$ be the feasible responses of the algorithm

B. Then, define $(R'_{i+1}, C'_{i+1}) = (R^B_{i+1}, C^B_{i+1}), \ldots, (R'_t, C'_t) = (R^B_t, C^B_t)$. Since the feasibility function is memoryless $\mathcal{F}_{i+j}(C^A_1, \ldots, C^A_i, C'_{i+1}, \ldots, C'_{i+j}) = 1$, for all $1 \leq j \leq t - i$. In addition, by the subadditivity of the load function it holds that

$$\begin{aligned}\text{cost}(\mathcal{R}^A_1, \ldots, \mathcal{R}^A_i, \mathcal{R}'_{i+1}, \ldots, \mathcal{R}'_t) &\leq \text{cost}(\mathcal{R}^A_1, \ldots, \mathcal{R}^A_i) + \text{cost}(\mathcal{R}'_{i+1}, \ldots, \mathcal{R}'_t) \\ &\leq \text{cost}(\mathcal{R}^A_1, \ldots, \mathcal{R}^A_i) + \text{cost}(\mathcal{R}^B_1, \ldots, \mathcal{R}^B_t),\end{aligned}$$

where the second inequality is by monotonicity of the load function. □

C Online Metric Oblivious Probabilistic Tree Embedding

We first give some basic definitions, and then proceed to the online constructions.

C.1 Probabilistic Partition and Hierarchical Probabilistic Partition

For a metric space V, for any $\Delta > 0$, a Δ-bounded probabilistic partition is a distribution $\mathcal{P}$ over partitions P of V, such that $P = \dot\cup C_j$, $C_j \subset V$, and $\text{diam}(C_j) \leq \Delta$. For a partition P, let $P(x)$ denote the cluster that contains x. Probabilistic partition $\mathcal{P}$ has padding parameter $\gamma > 0$ if for all $x \in V$ and for all $0 < \delta < 1$,

$$\Pr_{P \sim \mathcal{P}} [B(x, \delta\Delta/\gamma) \not\subseteq P(x)] \leq \delta.$$

Padding parameter is a well studied notion [LS91, KPR93, Bar96, FRT04]. For our construction we use the (offline) probabilistic partition of [ABN06], which is based on [Bar96]. They partition a given V into Δ-bounded clusters iteratively: For an unclustered $v \in V$ pick a random radius r from the distribution given by $p(r) = \frac{\chi^2}{1-\chi^{-2}} \frac{8 \ln \chi}{\Delta} \chi^{-\frac{8r}{\Delta}}$, for $r \in [\Delta/4, \Delta/2]$ and a parameter $\chi \geq 2$; Add a new cluster defined by the intersection of the ball $B(v, r)$ with the still unclustered points. Being more precise, the unclustered point v_j at the iteration j is chosen in the very specific way that depends on χ_j - the *local growth rate* parameter at the iteration j, see [ABN06] for details. The radius r_j at the j-th iteration is chosen according to $p(r)$ with $\chi = \chi_j$. However, inspecting their analysis more carefully, it can be noted that choosing v_j arbitrarily among the unclustered points and choosing r_j as before, with some χ_j that satisfy $\sum_{1 \leq j \leq t} \chi_j^{-1} \leq 1$, for t being the number of clusters that can be obtained by this construction, would imply a random partition with the same padding property. Let $\hat{\mathcal{P}}$ denote the random partition constructed in this way, and let $\{C_j\} \subset \hat{\mathcal{P}}$ denote its clusters. The following technical lemma was implicitly proved in [ABN06](Lemma 5):

Lemma C.1. *Let P be a randomly constructed partition, as described above. For any $x \in V$ consider the iteration j of the construction, at which $P(x) = C_j$. Then, for any $1/2 \leq \theta < 1$:*

$$\Pr_{P \sim \hat{\mathcal{P}}} \left[B\left(x, \frac{\ln(1/\theta)}{32 \ln \chi_j}\right) \not\subseteq P(x) \right] \leq (1 - \theta) \left(1 + \theta \sum_{l=1}^{t} 1/\chi_l \right),$$

where $\chi_j \geq 2$ is the parameter used to pick the radius at j-th iteration and t is the number of clusters in P.

Next we define the notion of a *hierarchical* probabilistic partition. For a given finite metric space V, let $A \leq B$ be two integers and consider a set of distance scales $\{\Delta_j | A \leq j \leq B\}$, such that $\Delta_A = d_{\max}(V)$, $\Delta_B = d_{\min}(V)$ and for $A \le j < B$, $\Delta_{j+1} = \Delta_j/\mu$. A hierarchical probabilistic partition $\mathcal{H}$ of V with parameter $\mu > 1$ is a collection of nested Δ_j-bounded probabilistic partitions $\mathcal{P}_j$ of V, for $A \le j \le B$. The partitions are nested, namely, each cluster C of each partition $P_j \sim \mathcal{P}_j$ is (randomly) partitioned by clusters $P_{j+1}[C]$ of each partition $P_{j+1} \sim \mathcal{P}_{j+1}$. If for each cluster $C \in P_j$, for each $P_j \sim \mathcal{P}_j$ the probabilistic partition $\mathcal{P}_{j+1}[C]$ (this is the probabilistic partition $\mathcal{P}_{j+1}$ induced on the cluster C) has padding parameter γ_j then the padding parameter of the whole hierarchy is defined by $\gamma(\mathcal{H}) = \sum_{A \le j \le B} \gamma_j$.

In [Bar96], among other results, Bartal showed that:

Theorem C.2 (Theorem 13, [Bar96])**.** *Given a hierarchical probabilistic partition $\mathcal{H}$ of V with parameter $\mu > 1$, one can construct a randomized embedding into an μ-HST tree, with expected distortion $O(\mu \cdot \gamma(\mathcal{H}))$.*

Thus, in what follows, we describe an online randomized algorithm that maintains a hierarchical partition H of the current point set X_i, with padding parameter $O(\mu \log_\mu \Phi(X_i) \log |X_i|)$. For each relevant scale of distances in the current point set, there is a bounded partition for this scale, which is maintained online as well. The algorithm does not assume a prior knowledge on the underlying metric space, it rather uses only information on the current point set.

C.2 Online Δ-Bounded Probabilistic Partition

Since we aim at hierarchical structure of partitions, we present an online construction of a Δ-bounded partition of a subspace $Z \subseteq V$, the points of which are revealed one by one to the algorithm. For a given Δ, Algorithm 1 maintains a random Δ-bounded partition $P \in \mathcal{P}$ (sampled from a distribution of Δ-bounded partitions $\mathcal{P}$) for the current subspace Z. The partition P is a collection of clusters of the form $C(v, r)$, each cluster is represented by the pair (v, r), where $v \in Z$ is the center of the cluster and $r > 0$ is its radius. Particularly, $C(v, r)$ is the subset of all points in Z within distance at most r from v. When a new point $z \in V \setminus Z$ is given to the algorithm, P is (randomly) updated to be Δ-bounded partition of $Z \cup \{z\}$. At each time step the padding parameter of the partition is $O(\log |Z|)$. At the beginning $X_0 = \emptyset$, $P = \emptyset$ and $t = 0$ ($0 \le t \le |Z|$ counts the current number of clusters in P). The argument P identifies a particular partition of a subspace Z that the algorithm maintains.

Algorithm 1 Δ-Bounded Online Probabilistic Partition, Δ-**BOPP**$\langle P \rangle$

Let $P = \dot\cup_{j=1}^{t} C(v_j, r_j)$ denote the current set of clusters in P. When a new point $z \in V \setminus Z$ arrives:

for all $1 \le j \le t$ **do**
  if $d(v_j, z) \le r_j$ **then**
    update $C(v_j, r_j) \leftarrow C(v_j, r_j) \cup \{z\}$;
    terminate
  end if
end for
Set $t \leftarrow t + 1$ and $\chi_t = 2t^2$. Pick r distributed according to $p(r) = \frac{(\chi_t)^2}{1-(\chi_t)^{-2}} \frac{8 \ln(\chi_t)}{\Delta} (\chi_t)^{-\frac{8r}{\Delta}}$, for $r \in [\Delta/4, \Delta/2]$, independently from the previous iterations. Set $C(v_t, r) \leftarrow \{z\}$. Update $P \leftarrow P \cup C(v_t, r)$.

Lemma C.3. *Let $P \in \mathcal{P}$ be a random partition of Z constructed by Algorithm 1 in $|Z|$ steps (starting from $Z = \emptyset$ and revealing points of Z one by one). The padding parameter of $\mathcal{P}$ is* $O(\log |Z|)$.

Proof. Note that after Z steps of the algorithm, the randomly constructed partition P is distributed by the same distribution as if it was constructed offline, over Z given upfront. For any $x \in Z$ and any $0 < \delta < 1$, we show that $\Pr_{P\sim\mathcal{P}}[B(x, \delta\Delta/(c \cdot \log |Z|)) \not\subseteq P(x)] \leq \delta$, for some constant c. By Lemma C.1, for any $1/2 \leq \theta < 1$, we have:

$$\Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\ln(1/\theta)}{32\ln\chi_j}\right) \not\subseteq P(x)\right] \leq (1-\theta)\left(1 + \theta\sum_{l=1}^{t} 1/\chi_l\right),$$

where j is such that $P(x) = C(v_j, r_j)$, and t is the number of clusters in P. Since $t \leq |Z|$ and since for all $1 \leq l \leq t$, $\chi_l = 2l^2$, it holds that:

$$\sum_{l=1}^{t} 1/\chi_l = \sum_{l=1}^{t} 1/(2l^2) < \frac{\pi^2}{12} < 1.$$

Therefore,

$$\Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\ln(1/\theta)}{32\ln\chi_j}\right) \not\subseteq P(x)\right] \leq 1 - \theta^2.$$

Letting $\delta = 1 - \theta^2$ results in

$$\Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\ln(1/(1-\delta))}{64\ln\chi_j}\right) \not\subseteq P(x)\right] \leq \delta.$$

Since for any $\delta > 0$, $\ln(1/(1-\delta)) \geq \delta$ it holds that $B\left(x, \frac{\delta}{64\ln\chi_j}\right) \subseteq B\left(x, \frac{\ln(1/(1-\delta))}{64\ln\chi_j}\right)$, implying

$$\Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\delta}{64\ln\chi_j}\right) \not\subseteq P(x)\right] \leq \Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\ln(1/(1-\delta))}{64\ln\chi_j}\right) \not\subseteq P(x)\right] \leq \delta.$$

Finally, since for all clusters $C_j \in P$, $\chi_j \leq 2|Z|^2$, it holds that $\frac{\delta}{64\ln\chi_j} \geq \frac{\delta}{c\cdot\ln(2|Z|^2)}$, for some constant $c > 0$, implying that

$$\Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\delta}{c\cdot\ln(2|Z|^2)}\right) \not\subseteq P(x)\right] \leq \Pr_{P\sim\mathcal{P}}\left[B\left(x, \frac{\delta}{64\ln\chi_j}\right) \not\subseteq P(x)\right] \leq \delta,$$

which completes the proof. □

C.3 Online Hierarchical Probabilistic Partition

Algorithm 2 maintains a random hierarchical partition H of the current terminal set X_i, with parameter $\mu > 4$. The hierarchy H is randomly sampled from a distribution $\mathcal{H}$. The scale set of this hierarchical partition is $\{\Delta_j | A \leq j \leq B\}$, where $A < B$ are some integers. At each time step i, the scales satisfy the following properties: $\Delta_A \geq 4 \cdot d_{\max}(X_i)$, $\Delta_B \leq \frac{4d_{\min}(X_i)}{\mu}$, and for all $A \leq j < B$, $\Delta_{j+1} = \Delta_j/\mu$. The number of scales is $B - A = \lceil \log_\mu \Phi(X_i) \rceil + 1$.

Essentially, the algorithm maintains randomly constructed Δ_j-bounded partitions P_j, for each scale Δ_j, such that the clusters of the partition P_{j+1} are Δ_{j+1}-bounded random partitions of the clusters of P_j. Moreover, for each cluster C of the partition P_j the random partition $\mathcal{P}_{j+1}[C]$ of C has padding parameter $O(\log|C|) = O(\log i)$. The algorithm has to maintain the current set of the relevant distance scales: When a new terminal x_i arrives the algorithm checks whether the aspect ratio has increased by at least a constant factor of μ with respect to the current upper or lower scale level. If so, it adds new, either top or bottom, scales to H. While creating new scales, the algorithm keeps the structure of H to be nested. The largest scale is defined such that its only cluster contains all the points of the current set X_i. Next, algorithm adds x_i to the relevant partitions of all scales in H, using Algorithm 1, while keeping the partitions to be nested as well. Particularly, for each cluster C of a partition of scale j there is a partition of it in level $j+1$, denoted by $P_{j+1}[C]$, which is maintained online by algorithm Δ_{j+1}-$\mathbf{BOPP}\langle P_{j+1}[C]\rangle$.

At the beginning $X_2 = \{x_1, x_2\}$, $A = 0, B = 1$, $\Delta_A = 4d(x_1, x_2)$, $\Delta_B = \Delta_A/\mu$. Set $P_A, P_B \leftarrow \emptyset$. Then, update these partitions as follows: apply $P_A \leftarrow \Delta_A$-$\mathbf{BOPP}\langle P_A\rangle$ on x_1 to add it to the partition P_A , and then apply Δ_A-$\mathbf{BOPP}\langle P_A\rangle$ on x_2 to add it to P_A. After these steps P_A contains one cluster, that contains both x_1 and x_2. Similarly, apply $P_B \leftarrow \Delta_B$-$\mathbf{BOPP}\langle P_B[P_A(x_1)]\rangle$ (recall that $P_A(x_1)$ denotes the cluster in partition P_A that contains x_1) on x_1 to add it the partition of the cluster $P_A(x_1)$ at level B, and then apply Δ_B-$\mathbf{BOPP}\langle P_B[P_A(x_2)]\rangle$.

Observation C.4. *We observe that Algorithm 2 is a proper online algorithm for maintaining hierarchical partition, i.e., updating the hierarchical partition is made in a way that does not split the previously constructed clusters: at each time step i each cluster that has been already created either stays unchanged or receives the new point.*

Lemma C.5. *Let H be a random collection of partitions that was constructed by Algorithm 2 when applied on a terminal set X_i of size i, and let $\mathcal{H}$ denote the distribution of H. Then, $\mathcal{H}$ is a hierarchical probabilistic partition and its padding parameter is bounded by $O(\log_\mu \Phi(X_i)\log i)$.*

Proof. First, note that for every $H \sim \mathcal{H}$ the partitions of all scales are nested by construction. Every $H \sim \mathcal{H}$ has a scale set such that $\Delta_A \leq 4\mu d_{\max}(X_i)$ and $\Delta_B \leq 4d_{\min}/\mu$, with the scales decreasing exactly by a factor of μ. The number of the scales after i steps is $O(\log_\mu \Phi(X_i))$. Each such random partition in H has been constructed using online algorithm Algorithm 1, implying that each probabilistic partition $\mathcal{P}_{j+1}[C]$, for each cluster $C \in P_j \sim \mathcal{P}_j$ has padding parameter $O(\log|C|) = O(\log i)$. This completes the proof. □

C.4 Online Probabilistic Tree Embedding

Proof of Theorem 1.3: In what follows we describe the construction of the metric-oblivious tree embedding. The extension of the embedding to V is presented in Claim C.7.

In [Bar96] Bartal gave a natural algorithm to construct an HST tree T (in the labeled tree representation) from a given hierarchical partition H of a metric space X. Let $P_A, \ldots, P_B$ denote the nested partitions of H. Recall that we assumed that P_A contains all the points of X as its only cluster. The root r of T is defined to have label Δ_A. Let $T_1, \ldots, T_s$ be the HST trees, recursively constructed for s clusters of the partition P_{A+1}. Connect $T_1, \ldots, T_s$ to the root r as direct children. Note that the points of X are the leaves of T, and that T is indeed a μ-HST tree.

Thus, using the above construction on a hierarchical partition H randomly generated by Algorithm 2, when applied on the current terminal set X_i, we obtain the appropriate μ-HST tree T_i.

It remains to ensure that we can keep the tree updating process in online manner, i.e., the tree of a time step i is a subtree of the tree constructed in the time step $i+1$, which is true by Observation C.4. To conclude the proof of the theorem, we apply Lemma C.5 to Theorem C.2. □

Remark C.6. *We note that the resulting tree T_i, constructed from a hierarchical partition H of X_i, can contain "redundant" paths: these are chain-paths from a node v to u, created from partitions that were added to H when the aspect ratio of the current set has changed (either increased or decreased). We can compress these chains (each longest chain) to one edge without changing the metric defined by the tree. In addition, the compressed tree will be a proper μ-HST tree as well, with the number of nodes bounded by $O(|X_i|)$.*

Algorithm 2 Online Hierarchical Probabilistic Partitions

When a new terminal x_i arrives:
if $d_{\max}(X_i) > \frac{\mu}{4}\Delta_A$ **then**
  Let $l = \lceil \log_\mu(4d_{\max}(X_i)/\Delta_A) \rceil$, and update $A \leftarrow A - l$. Define $P_{A-1} = \{X_i\}$.
  for all $A \le j < A + l$ **do**
    create new $\Delta_j = \Delta_{A+l} \cdot \mu^{(A+l-j)}$- bounded partition $P_j = \emptyset$;
    for all $1 \le s \le i-1$ **do**
      apply Δ_j-**BOPP**$\langle P_j[P_{j-1}(x_1)] \rangle$ on input x_s.
    end for
  end for
end if
if $d_{\min}(X_i) < \frac{\mu}{4}\Delta_B$ **then**
  Let $l = \lceil \log_\mu(\Delta_B/(4d_{\min}(X_i))) \rceil + 1$, and update $B \leftarrow B + l$.
  for all $B - l < j \le B$ **do**
    create new $\Delta_j = \Delta_{B-l}/\mu^{j-B+l}$ bounded partition $P_j = \emptyset$.
    for all $1 \le s \le i-1$ **do**
      apply Δ_j-**BOPP**$\langle P_j(P_{j-1}(x_s))] \rangle$ on input x_s.
    end for
  end for
end if
Create $P_{A-1} = \{X_i\}$.
For all $A \le j \le B$: apply Δ_j-BOPP$\langle P_j[P_{j-1}(x_i)] \rangle$ on input x_i.

Claim C.7. *Let V be any metric space and $X = \{x_1, \ldots, x_k\} \subseteq V$ is a sequence of terminals. The online probabilistic metric-oblivious embedding f into a μ-HST T of Theorem 1.3 is a fully extendable online tree embedding.*

Proof. For each $1 \le i \le k$, let $f_i \colon X_i \to T_i$ denote the embedding of step i, where T_i is the random μ-HST tree constructed by the algorithm of Theorem 1.3. The tree T_i is constructed from the corresponding hierarchical partition H of the terminal set X_i. To define the extension function $F_i : V \to T_i$, we use the shortest path representation of the metric T_i. We first construct the extension of f_i, $F_i : V \to T_i \cup \ell_i$, where ℓ_i is an additional leaf added to T_i as a child of the root. Particularly, F_i either maps each $v \in V \setminus X_i$ to some node of T_i or to the newly added leaf ℓ_i. The construction is as follows: for all $v \in V \setminus X_i$ simulate Algorithm 2 on step i to find a cluster to which

v would be added if the partition at each scale was done on the underlying space V. Particularly, going recursively from the topmost scale of H to the lowest, for each cluster $C(u, r)$ in partition P_j, according to the order of construction, check whether v belongs to it, i.e. whether $d(u, v) \leq r$. Let $j \geq 1$ be the first level such that v does not belong to any cluster of the partition P_j. If $j = 1$, let $F_i(v) = \ell_i$, where ℓ_i is the leaf connected directly to the root of T_i, with edge length being $\Delta_A/2$. Otherwise, if $j > 1$, then let u be the center of the cluster in partition P_{j-1} that contains v. By the construction of the tree embedding f_i, there is an internal node $\hat{u}$ on the tree T_i that corresponds to u. Let $F_i(v) = \hat{u}$.

Note that any two points that were mapped to ℓ_i or to the same internal node on the tree have expansion 0. Further, for any point $v \in V$ such that there is a partition P_j of X_i, with the cluster $C(u, r)$ satisfying $d(u, v) \leq r$, for any $w \in V$, the probability that the ball $B_V(v, d(v, w))$ cut by the cluster $C(u, r)$ is bounded by $\log |X_i| d(v, w)/\Delta_j$, where Δ_j is the diameter of $C(u, r)$. This implies that the expected expansion of a pair of points $v, w \in V$ that were not mapped to the same node is bounded by $O(\log |X_i|) = O(\log i)$, as required.

To define an extension H_i of f_i^{-1} we again use the shortest path representation of HST's. The extension $H_i : T_i \to V$ is defined as follows. Going from the leaves of T_i up to the root, for each node $v \in T_i$: if $H_{i-1}(v)$ is already defined then let $H_i(v) = H_{i-1}(v)$, otherwise, if v is a leaf then let $H_i(v) = f_i^{-1}(v)$ and for each internal non-leaf node v let $H_i(v) = H_i(u)$ where u is any child of v. Note that by the construction H_{i+1} extends H_i, since T_i is a subtree of T_{i+1}. Also, since f_i is non-contractive on the terminal set X_i, for any two terminals $x_j \neq x_l$ we have $d(x_j, x_l) \leq d_{T_i}(f_i(x_j), f_i(x_l))$, i.e., $d(H_i(f_i(x_j)), H_i(f_i(x_l))) \leq d_{T_i}(f_i(x_j), f_i(x_l))$. Next we show that H_i is non-expansive on any two internal nodes $u \neq v \in T_i$. Let $f_i(x_j)$ be any leaf in the subtree rooted on v, and $f_i(x_l)$ be any leaf in the subtree rooted on u. Since T_i is a μ-HST it holds that $(1/4) d_{T_i}(f_i(x_j), f_i(x_l)) \leq d_{T_i}(u, v)$, implying that H_i is non-expansive up to a factor of 4. Scaling up the edge costs of T_i by 4 gives a non-expansive embedding. □

By the above construction and Remark C.6 we have:

Observation C.8. *The extension F_k of the online embedding $f_k : X_k \to T_k$ maps the points of V into an HST tree with $O(k)$ nodes.*