Improved Embeddings of Graph Metrics into Random Trees*

Kedar Dhamdhere Anupam Gupta Harald Räcke
Computer Science Department,
Carnegie Mellon University,
Pittsburgh PA 15213.

Abstract

Over the past decade, numerous algorithms have been developed using the fact that the distances in any n-point metric (V, d) can be approximated to within $O(\log n)$ by distributions $\mathcal{D}$ over trees on the point set V [3, 10]. However, when the metric (V, d) is the shortest-path metric of an edge weighted graph $G = (V, E)$, a natural requirement is to obtain such a result where the support of the distribution $\mathcal{D}$ is only over *subtrees* of G. For a long time, the best result satisfying this stronger requirement was a $\exp\{\sqrt{\log n \log\log n}\}$ distortion result of Alon et al. [1]. In a recent breakthrough, Elkin et al. [9] improved the distortion to $O(\log^2 n \log\log n)$. (The best lower bound on the distortion is $\Omega(\log n)$, say, for the n-vertex grid [1].)

In this paper, we give a construction that improves the distortion to $O(\log^2 n)$, improving slightly on the EEST construction. The main contribution of this paper is in the analysis: we use an algorithm which is similar to one used by EEST to give a distortion of $O(\log^3 n)$, but using a new probabilistic analysis, we eliminate one of the logarithmic factors. The ideas and techniques we use to obtain this logarithmic improvement seem orthogonal to those used earlier in such situations—e.g., Seymour's decomposition scheme [4, 9] or the cutting procedures of CKR/FRT [5, 10], both which do not seem to give a guarantee of better than $O(\log^2 n \log\log n)$ for this problem. We hope that our ideas (perhaps in conjunction with some of these others) will ultimately lead to an $O(\log n)$ distortion embedding of graph metrics into distributions over their spanning trees.

1 Introduction

The area of finite metric spaces and their embeddings into "simpler" metrics lies in the intersection of the areas of mathematical analysis, computer science and discrete geometry. This area, loosely called *metric embeddings*, has seen a tremendous amount of activity in the past decade, fueled by the many algorithmic applications of these techniques. Many such examples can be found in the survey by Indyk [13].

Some of the results in metric embeddings that have had the most impact have been on *approximating metric spaces by distributions over dominating trees*. Formally, given a metric space $M = (V, d)$, the goal is to find a distribution $\mathcal{D}$ over edge-weighted trees on the vertex set V which satisfy the following conditions: (a) for any tree T in the support of $\mathcal{D}$, we want that $d(x, y) \le d_T(x, y)$, i.e., the distances in T *dominate* those in M; and (b) the expected distance $E_{T\in\mathcal{D}}[d_T(x, y)] \le \alpha d(x, y)$ for all $x, y \in V$. We call this quantity α the *distortion* of the embedding $M \hookrightarrow D$, and the goal is to find the smallest α possible. Given an optimization problem Π defined on a metric space, where the objective function is a linear function of the distances, it is easy to show that obtaining a β-approximation to the problem Π on tree metrics immediately gives a randomized $\alpha \cdot \beta$-approximation algorithm for Π on the metric M.

A succession of influential papers [1, 3, 4, 6, 7, 10] recently culminated in the result that one can embed any n-point metric M into a distribution of trees with a distortion of $\alpha = O(\log n)$, and that $O(\log n)$ is the best possible guarantee for general metrics. (Better results are known for some special classes of metrics [12, 8].) These results form the basis for the best approximation algorithms known for many problems: e.g., buy-at-bulk network design [2], group Steiner tree [11], metric labeling [14], etc.

A Stronger Requirement: Random Subtrees. In some situations, one is given a edge-weighted graph $G = (V, E)$, and the metric M is the shortest-path metric d_G of the graph with respect to these edge-weights. In such cases, it is natural to require that the distribution $\mathcal{D}$ have only *subtrees* of the graph G in its support. (Note that this is a more general problem: the metric $M = (V, d)$ can be obtained by taking the

*{kedar, anupamg, harry}@cs.cmu.edu. The research of the first and third authors supported in part by NSF grant no. CCR-0122581 (The ALADDIN project), and the research of the second author partly supported by an NSF CAREER award CCF-0448095, and by an Alfred P. Sloan Fellowship.

complete graph on V, and assigning a weight of $d(u,v)$ to the edge (u,v).) For a long time, the best value of α known for this case was $\exp\{\sqrt{\log n \log\log n}\}$, which was given by a paper of Alon et al. [1]. In a recent breakthrough, Elkin et al. [9] gave a construction with a distortion of $\alpha_{EEST} = O(\log^2 n \log\log n)$; for brevity, we henceforth refer to this as the *EEST construction*. The bound α_{EEST} is not known to be tight, as the best lower bound for this case is also $\Omega(\log n)$, e.g., for the n-vertex grid [1].

1.1 Our results. The main quantitative result of this paper is the following:

THEOREM 1.1. *Any n-point metric metric can be embedded into a distribution over spanning trees with distortion $\alpha = O(\log^2 n)$.*

This result improves upon the EEST bound (by a factor of $O(\log\log n)$). The contribution of the paper, however, is not merely in this quantitative improvement, but in the new techniques and ingredients of our analysis, which we briefly sketch now.

Loosely, most constructions of embeddings into tree distributions have been based on the following idea: given a graph of diameter D, partition it into subgraphs of diameter about $D/2$ (while ensuring that the probability of an edge (u,v) being cut is $\approx p_{uv}(D) \doteq \frac{\log n}{D/2} d(u,v)$), recursively build random trees for these subgraphs, and then combine these trees together to get a tree for the entire graph. For the case of non-subtrees, one can perform this combination step whilst ensuring the diameter of the new tree is only $O(D)$, and hence the expected increase in the (u,v) distance due to the partition (and combination) step at diameter D is

$$\approx \frac{\log n}{D/2} d(u,v) \cdot O(D) = O(\log n)\ d(u,v). \qquad (1.1)$$

Since there are $O(\log(\mathrm{diam}(G)))$ levels of recursion, each of which resulting in an increase of (1.1) in the expected (u,v) distance, it follows that the final distortion is $O(\log(\mathrm{diam}(G)) \log n)$. Indeed, this is essentially Bartal's scheme to get $\alpha = O(\log^2 n)$.[1] This non-subtree result was subsequently improved by analyses that did account for increases more carefully, and did not pay $O(\log n)$ at each of the $O(\log n)$ levels of the recursion: Bartal [4] (see also [7]) obtained $\alpha = O(\log n \log\log n)$, and Fakcharoenphol et al. [10] obtained $\alpha = O(\log n)$.

The Trouble with Subtrees. However, now requiring the random tree to be a subtree of G makes things more complicated: since we cannot combine two subtrees by connecting "non-edge" pairs in $(V \times V) \setminus E$, it becomes difficult to ensure that the eventual diameter of the random subtree is bounded above by $O(D)$. In order to do this two things are important. Firstly it is necessary to ensure that the cluster that result from the partitioning for diameter D form a tree of constant diameter (and—as previously—each clustger should have diameter $\leq D/2$). Then directly after a partitioning step it is possible to connect a pair u and v by a path of length $O(D)$ that uses only tree edges between clusters (see Figure 1). In a second step however it is important to ensure that the length of such a path does not increase too much in later levels of the recursion (see Figure 1 for an example where the length of the u-v path changes in the recursion). The most convenient way to show this is to show that clusters generated in a partioning step do not increase their diameter by more than a constant factor due to the recursion.

The simpler EEST construction ensures this by a partitioning scheme that cuts more aggressively—their "cone decompositions" cut an edge with probability $\approx \frac{O(\log^2 n)}{D/2} d(u,v) = O(\log n) \cdot p_{uv}(D)$, and use this to bound the eventual diameter of the tree by $O(D)$.

Since the cutting probability increases by a factor of $O(\log n)$, the expected increase in the (u,v) tree-distance due to the partition (and combination) step at diameter D is now $O(\log^2 n)\ d(u,v)$, by a calculation similar to that in (1.1). And since there are $O(\log n)$ levels, one gets a total distortion of $O(\log^3 n)$ by summing over all levels. Note that this loss of an $O(\log^2 n)$ factor within a fixed level of the recursion makes it unlikely that the better accounting schemes of Bartal, and of Fakcharoenphol et al. would yield an $\alpha = o(\log^2 n)$ result. In fact, these ideas [4, 7, 10] only seem to get the distortion down to $\alpha_{EEST} = O(\log^2 n \log\log n)$. (See Section 7 for a detailed discussion of these schemes.)

Our Technical Ideas. In this paper, we revert back to decompositions which cut edge (u,v) at diameter D with probability $\approx p_{uv}(D)$, and hence lose only $O(\log n)$ distortion at each level of the recursion. Our approach is to ensure that the distance between u and v in the random subtree is bounded by $O(D)$ *only with high probability*. The first idea is based on the fact that the standard graph partitioning scheme of Bartal not only decomposes a graph into subgraphs of diameter $D/2$ w.h.p., but the *expected* diameter is much smaller—only $D \cdot O(1/\log n)$. From this it follows that in expectation when using the Bartal scheme the diameter of a component increases by a factor of only $O(1 + \frac{1}{\log n})$ in a recursion level. If now the behavior at

[1] If the diameter $\mathrm{diam}(G)$ is not polynomial, an additional trick is required to get $O(\log^2 n)$—however, this is tangential to our current story, so let us assume for now that the diameter of G is polynomially bounded in n.

each level is close to the expectation, then the diameter of the random tree over $L = O(\log n)$ levels would only be $D(1 + O(1)/\log n)^L = O(D)$. While this intuition is correct, proving this formally requires us to develop some concentration bounds.

In summary, we give embeddings of graphs into their random subtrees, where we improve the distortion from the previous best of $O(\log^2 n \log\log n)$ down to $O(\log^2 n)$. The techniques we develop for our result seem to be orthogonal to those previously used by [4, 7, 10] to improve the distortion for random tree embeddings (not into subtrees); while we do not see an immediate way to combine our techniques with these previous techniques to get $o(\log^2 n)$ distortion, we surmise that the techniques we present will prove useful in achieving a $O(\log n)$ distortion result for random subtrees.

2 Notation and Preliminaries

In the following we use $G = (V, E)$ to denote an unweighted graph and we use $d(u, v)$ to denote the shortest path distance between two nodes u,v in V. Note that the restriction to unweighted graphs is only done in order to state the main concepts of the paper as clearly and simple as possible. All results hold for weighted graphs, as well.

In [9] Elkin et al. introduce the concept of a star-decomposition of a graph G.

DEFINITION 2.1. (STAR-DECOMPOSITION) *A star-decomposition of a graph G with a designated root node r_0 is a set of disjoint connect components of $G_0 = (V_0, E_0), \ldots, G_k = (V_k, E_k)$ together with a collection of root nodes $r_0, \ldots, r_k$ such that $r_i \in V_i$ for all $0 \le i \le k$ and each r_i, $1 \le i \le k$ has a neighbor in V_0.*

They use an algorithm for finding a star-decomposition to recursively construct a spanning tree of the graph G in the following way.

Construct Spanning Tree:

- Construct a star-decomposition of G.
- Let G' denote the (multi-)graph obtained from G by merging each connected component G_i of the star-decomposition into a single super-node, and let v_0 denote the node in G' that corresponds to component G_0. Select a spanning tree T' on G' such that T' is a star with center node v_0.
- Recursively construct spanning trees for each component G_i with root r_i.
- The set of all selected edges forms a spanning tree of G.

3 Finding a star-decomposition

We use the following randomized algorithm to create a star-decomposition of a given graph G with root r_0 and radius Δ. This algorithm can be applied recursively as described in the above scheme to obtain a spanning tree of G. In the following section we will prove that the spanning tree that results from this process has low average stretch. The algorithm first creates the component G_0 using a random radius cut around center r_0.

Construct component G_0 (forward cut):

Choose a radius γ uniformly at random from the interval $[\Delta/4, \Delta/2]$. Cut all edges at distance γ from the root r_0 (more formally: cut edges $(u, v) \in E$ for which $d(r, u) \le \gamma \le d(r, v)$). Let V_0 denote the connected component that contains r_0.

Let $x_1, \ldots, x_s$ denote the vertices in $V \setminus V_0$ that had a neighbor in V_0 (which is now cut away). We call these nodes *portal nodes*. In order to select the components $V_1, \ldots, V_k$ we successively cut pieces from the remaining graph $G \setminus V_0$. We will again use random radius cuts to do this; however we first introduce a new distance function on the node set $X = V \setminus V_0$ and the random radius will then be interpreted with respect to this distance function.

For the definition of the new distance function we replace each edge in $G \setminus V_0$ by two directed edges in opposite direction. We define the length $\ell(u, v)$ of such a directed edge (u, v) as

$$\ell(u,v) = \begin{cases} 1 & \text{if } d(r_0, v) = d(r_0, u) - 1 \\ 1 & \text{if } d(r_0, v) = d(r_0, u) \\ 0 & \text{if } d(r_0, v) = d(r_0, u) + 1 \end{cases}$$

By this length-definition we get a shortest path distance on X which we call the *backward-edge distance* in the following (the distance from x to y counts how often an edge has to be used in backward or sidewards direction according to distance from r_0 in order to reach y from x). Using the new distance function we construct the components $G_1, \ldots, G_k$ in the following way. (Note that we only used directed edges to define the new distance function on X. In the following all edges are undirected again.)

Construct components $G_1, \ldots, G_k$ (backward cuts):

As long as there exists a portal node x_i that is not yet assigned to any component, construct a new component as follows.

Choose a random radius γ with distribution $\Delta \cdot \mathrm{Exp}(\lambda)$ for $\lambda = \Theta(\log n)$ and cut all edges at

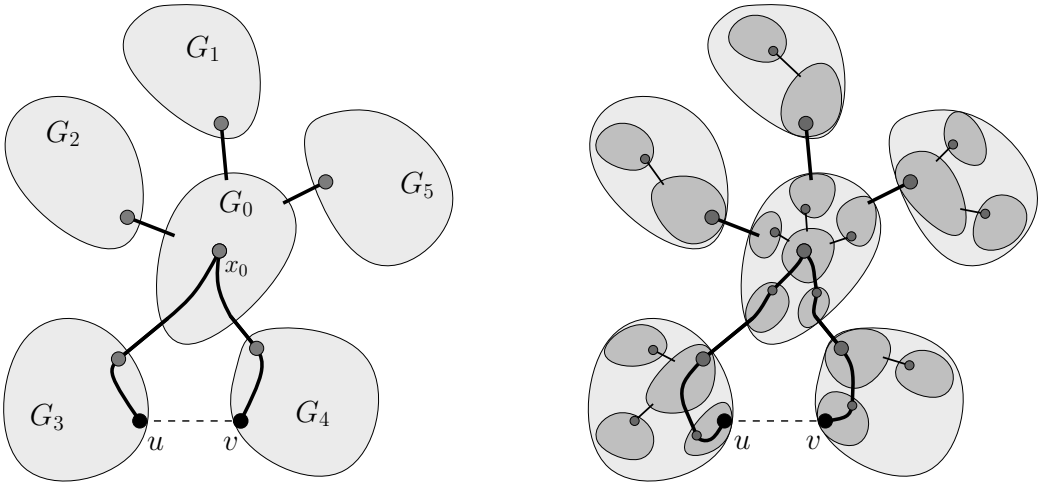

Figure 1: Recursive application of the star-decomposition. Note that the initial u-v path increases in length due to later levels of the recursion.

backward-edge distance γ from x_i to create a new component (we cut edge (u, v) if $\ell(x_i, u) \leq \gamma < \ell(x_i, v)$). The portal x_i becomes the root of the new component.

Note that the definition of backward-edge distance guarantees that whenever there is a node that is not yet assigned to a component then there is a portal node that is not assigned to a component. This property is in fact the reason behind the definition of the backward-edge distance.

4 Basic Properties

In this section we prove some basic properties about the randomized star-decomposition algorithm introduced in the previous section and about the spanning tree construction based on this algorithm. The actual analysis of the average stretch of the spanning tree is done in Section 5.

The first two claims analyze the probability for an edge to be cut in the froward-cut or backward-cut, respectively, of the star-decomposition.

CONDITION 4.1. *The probability that an edge is cut by the forward cut is at most* $\frac{4}{\Delta}$.

Proof. Consider an edge $e = (u, v)$. If v is closer than $\Delta/4$ or farther than $\Delta/2$ from the root r_0, then (u, v) cannot be cut by the forward cut step. Otherwise e will be cut iff $\gamma \in [d(r, u), d(r, v)]$. Since γ is chosen uniformly at random from the interval $[\Delta/4, \Delta/2]$, this happens with probability at most $4/\Delta$. ∎

CONDITION 4.2. *The probability that an edge is cut by a backward cut is at most* $O(\log n/\Delta)$.

Proof. Recall that in the backward cut step, we create a component V_i by choosing a random radius R_i according to distribution $\Delta \cdot \text{Exp}(\lambda)$ and cutting all edges at backward-edge distance R_i from the root r_i.

Consider an edge $e = (u, v)$. This edge is cut, if exactly one of u and v is inside the ball V_i. Without loss of generality, assume that u is closer (in backward-edge distance) to r_i than v. We consider three separate cases. If $R_i \geq \ell(r_i, v)$, then the edge (u, v) is inside the ball V_i and won't be cut. If $\ell(r_i, u) \leq R_i < \ell(r_i, v)$, then the ball V_i cuts the edge e. Finally, if $R_i < \ell(r_i, u)$, then the edge e is outside ball V_i and some other ball can cut it. Let p denote the probability that the edge e is cut. Then we have

$$p \leq \mathbf{Pr}[\ell(r_i, u) \leq R_i < \ell(r_i, v)] + \mathbf{Pr}[R_i < \ell(r_i, u)] \cdot p$$

By the triangle inequality we have $\ell(r_i, v) \leq \ell(r_i, u) + \ell(u, v)$. Let $d = \ell(r_i, u)$. Then

$$\begin{aligned} p &\leq \frac{\mathbf{Pr}[d \leq R_i \leq d + \ell(u, v)]}{1 - \mathbf{Pr}[R_i < d]} \\ &= \frac{\mathbf{Pr}[R_i \geq d \ \wedge \ R_i \leq d + \ell(u, v)]}{\mathbf{Pr}[R_i \geq d]} \\ &= \mathbf{Pr}[R_i \leq d + \ell(u, v) \mid R_i \geq d] \\ &= \mathbf{Pr}[R_i \leq \ell(u, v)] \\ &\leq 1 - e^{-\lambda \frac{1}{\Delta}} \leq \frac{\lambda}{\Delta} = O(\log n/\Delta) \ . \end{aligned}$$

The third step follows from the memoryless property of the exponential distribution and the fourth step follows since $\ell(u, v) \leq 1$. ∎

Combining the previous two claims we get the following simple corollary that forms one cornerstone for the analysis in Section 5.

COROLLARY 4.1. *The probability that an edge e is cut during the star-decomposition for a graph of radius Δ is less than $O(\log n/\Delta)$.*

The following claim allows us to bound the number of recursion levels needed by the recursive spanning tree construction based on our star-decomposition algorithm.

CONDITION 4.3. *With high probability the radius of component V_i is at most $\frac{7}{8}\Delta$.*

Proof. Let δ denote the length of a shortest path in G_i from root r_i to some other node $v \in V_i$. We show that with high probability $\delta \leq \frac{7}{8}\Delta$.

With high probability the random radius γ for the backward cut that created component G_i is at most $\Delta/16$. This means that all but $\Delta/16$ edges on the path from r_i to v increase the distance from the initial root node r_0. The node r_i is at least at distance $\Delta/4$ from r_0 because of the parameters in the definition of a forward cut. This means that v is at least at distance $\delta - 2\frac{\Delta}{16} + \Delta/4$ from r_i. Since this has to be less than Δ we get $\delta \leq \frac{7}{8}\Delta$. ∎

COROLLARY 4.2. *The star-decomposition algorithm has $O(\log n)$ levels of recursion.*

5 Analysis

A crucial term for analyzing the average stretch created when applying the above star-decomposition algorithm recursively to get a spanning tree is the probability that a given edge is cut by the decomposition algorithm. In the previous section we showed e.g. that this probability is $O(\log n/\Delta)$ for a star-decomposition on a graph of radius Δ.

If we only wanted to find a tree embedding with low average stretch, then this would be sufficient. We could create a parent node (which represents network G), add $k+1$ child-nodes (representing networks $G_0, \ldots, G_k$) and create a link of length Δ from each child to the parent node. Doing this recursively would result in a tree with expected distance of $O(\log^2 n)$ between the end-points u and v of an edge $(u,v) \in E$. This holds because if u and v are separated in this hierarchical decomposition on level Δ (i.e., the least common ancestor is a tree node that was created when decomposing a network of radius Δ), then their final distance in the tree will be $O(\Delta)$. Now, summing over the expected contribution of each level to the distance between u and v gives $O(\log^2 n)$. However, in our scenario when the final tree has to be a spanning tree of G, this is more involved. The problem is that we cannot immediately guarantee that a pair that is separated on level Δ will be at distance $O(\Delta)$ in the final tree. However, as captured by the following theorem, this guarantee is still still true, and implies

THEOREM 5.1. *Let (u,v) denote an edge that is separated in the recursive spanning tree construction by the star-decomposition of a sub-graph G with radius Δ. Then with high probability the distance between u and v in the final tree is $O(\Delta)$.*

Before we prove this result, let us see how it implies the main theorem of the paper.

THEOREM 5.2. *The spanning tree computed by recursively applying the star-decomposition algorithm has expected stretch $O(\log^2 n)$.*

Proof of Theorem 5.2. Fix an edge $e = (u,v)$. It is sufficient to show that the expected length of the final distance $d_k(u,v)$ between u and v is $O(\log^2 n)$. From Theorem 5.1 we know that if u and v are separated during a star-decomposition for a sub-graph of radius Δ, then the final distance between them is $O(\Delta)$, w.h.p. Note that in the following it is sufficient to consider events that only happen with high probability because the worst distance that an edge can get is n. This means that events that occur with probability only $\frac{1}{\text{poly}n}$ can be ignored as their contribution to the expected distance is low.

Now, we get that

$$\begin{aligned}\mathbf{E}[d_k(u,v)] &= \sum_{\text{levels } \Delta} \mathbf{Pr}[e \text{ is cut at level } \Delta] \cdot O(\Delta) \\ &\leq \sum_{\text{levels } \Delta} O(\frac{\log n}{\Delta}) \cdot O(\Delta) \\ &= O(\log^2 n) \ .\end{aligned}$$

∎

Proof of Theorem 5.1. Let G_i and G_j denote the sub-graphs of G that contain u and v respectively in the star-decomposition for G. As argued in Claim 4.3 both graphs G_i and G_j have radius less than $\frac{7}{8}\Delta < \Delta$, w.h.p. Let p_i and p_j denote the neighbor in V_0 of r_i and r_j, respectively. Furthermore, the distances $d(r_0, r_i)$ and $d(r_0, r_j)$ are less than $\Delta/2 + 1$, since the forward cut is made at distance at most $\Delta/2$.

The cuts for the star-decomposition of G increases the distance between u and v only to $d(u,r_i) + 1 + d(p_i, r_0) + d(r_0, p_j) + 1 + d(r_j, v) < 4\Delta + 2$. This means that right after the separation of u and v the distance between u and v is ok. However, deeper in the recursion it might happen that the distances $d(u,r_i), d(p_i,r_0), d(r_0,p_j)$ or $d(r_j,v)$ change due to additional cuts that are made while replacing $G_0, \ldots, G_k$ by spanning trees. The following lemma, whose proof appears below, shows that with high probability this increase is not too large.

LEMMA 5.1. *Let $G = (V,E)$ denote a sub-graph rooted at r that appears during the recursive spanning*

tree construction. The final distance between the root r and a node $v \in V$ is $O(d(r,v))$ w.h.p.

Since right after the star-decomposition of G the distance between u and v is $O(\Delta)$ and by the above Lemma the distance only changes by constant factors during the further recursion, this completes the proof of Theorem 5.1. ∎

Proof of Lemma 5.1. We can view the spanning tree construction algorithm for G as an iterative algorithm in the following way. The algorithm maintains a set of connected components $\mathcal{G}$ and a set E_T of tree edges. Initially, $\mathcal{G}$ contains only G and the set E_T is empty. In a round of the algorithm every component $H \in \mathcal{G}$ is replaced by the set of connected components obtained by applying a star-decomposition to H, and the edges (r_i, p_i) between the new components are added to E_T.

We now analyze how the length of an $r - v$ path changes over the course of this iterative process. At any time the shortest path node v and r consists of a set of tree-edges together with a set of sub-paths through components, where each sub-path goes from some node of the respective component to its root. Fix a round i. Let m denote the number of components that are currently visited by the $v-r$ path after this round, and let G_j denote the j-th such component. Further let v_j, $1 \leq j \leq m$ denote the vertex where the path enters G_j, and let r_j denote the last node in this component (note that $v_1 = v$ and $r_m = r$). We use Δ_j to denote the radius of component G_j, and for any two nodes $x, y \in V$ we use $d_i(x,y)$ to denote the distance between them after the i-th round.

We first analyze how the distance between v and r changes in the $(i+1)$-st round. For this we calculate how the length of a $v_j - r_j$ sub-path changes. Suppose that the length $d_i(v_j, r_j)$ of the sub-path is less than $\Delta_j/4$ before the $(i+1)$-st round. Then v_j and r_j will be contained in the same component after the round because the forward-cut will not cut the path. Hence, $d_{i+1}(v_j, r_j) = d_i(v_j, r_j)$ in this case.

Now, suppose that the forward-cut separates v_j from r_j. Let G' denote the new component that contains v_j and let r' denote the root of this component. The distance between v_j and r_j increases at most by twice the number of backward edges on the path from r' to v_j. This number is bounded by twice the backward-edge radius of the new component G' which is chosen randomly according to distribution $\Delta_j \cdot \text{Exp}(\lambda)$. Let R_j denote the backward-radius of G', and let $X_j = R_j/\Delta_j$. The total increase in the distance between v_j and r_j is bounded by $2\Delta_j X_j < 8 d_i(v_j, r_j) X_j$.

This holds for every j. Therefore, we can bound the distance $d_{i+1}(v,r)$ between v and r by

$$\begin{aligned} d_{i+1}(v,r) &\leq d_i(v,r) + \textstyle\sum_j 8 d_i(v_j, r_j) X_j \\ &\leq d_i(v,r)(1 + \textstyle\sum_j \alpha_j X_j) \ , \end{aligned}$$

where $\alpha_j = 8 d_i(v_j, r_j)/d_i(v,r)$. Note that $\sum_j \alpha_j \leq 8$. Let $\rho_{i+1} = d_{i+1}(v,r)/d_i(v,r)$ denote the relative increase of the $v - r$ distance in round $i+1$. We have $\rho_{i+1} \preceq 1 + \sum \alpha_j X_j$, and using Proposition 6.1 and Lemma 6.1 from the next section we can choose parameters $\beta = 16$ and $\gamma = 96/\log n$ such that $\rho_j \preceq 1 + \beta \cdot Y_{i+1} + \gamma$, where $Y_{i+1} \sim \text{Exp}(\lambda)$ is an exponentially distributed variable.

The final distance $d_k(v,r)$ for $k = O(\log n)$ can be bounded by

$$\begin{aligned} d_k(v,r) &= d(v,r) \cdot \textstyle\prod_{i=1}^{k} \rho_i \\ &\preceq d(v,r) \cdot \textstyle\prod_i (1 + \beta \cdot Y_i + \gamma) \\ &\leq d(v,r) \cdot \exp\left(\textstyle\sum_i (\beta Y_i + \gamma)\right) \\ &\leq d(v,r) \cdot \exp(96k/\lambda) \cdot \exp(\beta \textstyle\sum_i Y_i) \\ &\leq d(v,r) \cdot \exp(O(1)) \ , \ \text{w.h.p.}, \end{aligned}$$

where the last inequality follows from the tail bound proven in Lemma 6.2 in the next section. This finishes the proof of Lemma 5.1. ∎

6 Stochastic Domination and Tail Bounds

In this section we prove various lemmas that are used in the proof of Lemma 5.1. The first proposition is a standard result from probability theory and its proof is incorporated for completeness.

PROPOSITION 6.1. *The following facts hold:*

- *If $X \preceq Y$ and $Y \preceq Z$, then $X \preceq Z$.*
- *Let X, Y, and Z be independent r.v.s, with $Z \geq 0$. If $X \preceq Y$, then $XZ \preceq YZ$.*

Proof. Recall that $X \preceq Y$ holds iff $Pr[X \geq t] \leq Pr[Y \geq t]$ for all t. The first property follows from the fact that

$$\mathbf{Pr}[X \geq t] \leq \mathbf{Pr}[Y \geq t] \leq \mathbf{Pr}[Z \geq t]$$

To prove the second property, assume that $f(z)$ is the probability density function for the random variable Z. Then we have

$$\begin{aligned} \mathbf{Pr}[XZ \geq t] &= \int \mathbf{Pr}[Xz \geq t] f(z) dz \\ &\int \leq \mathbf{Pr}[Yz \geq t] f(z) dz = \mathbf{Pr}[YZ \geq t]. \end{aligned}$$

∎

The following lemma shows that a conic combination of exponentially, independently and identically distributed random variables can be dominated by a suitably scaled and shifted exponentially distributed variable with the same arrival rate.

LEMMA 6.1. *Let $X_1, \ldots, X_m \sim \text{Exp}(\lambda)$ be independent, exponentially distributed random variables with mean λ and let $\alpha_1, \ldots, \alpha_m$ denote m non-negative real numbers whose sum is bounded by α. Then for $\beta = 2\alpha$ and $\gamma = 2\frac{\alpha}{\gamma}$ we have*

$$\sum_i \alpha_i X_i \preceq \beta \cdot Y + \gamma \ ,$$

where $Y \sim \text{Exp}(\lambda)$ is again exponentially distributed. Equivalently,

$$\mathbf{Pr}[\textstyle\sum_i \alpha_i X_i \geq t + \gamma] \leq \mathbf{Pr}[\beta Y \geq t] \quad \textit{for all } t \geq 0 \ .$$

Proof. In order to prove the bound we first calculate the moment generating function $M(s)$ for $\sum_i \alpha_i X_i$. Recall that the moment generating function of a random variable Z is defined as $\mathbf{E}\left[e^{sZ}\right]$. Using this definition we get that the moment generating function M_i for $\alpha_i X_i$ is $M_i(s) = \frac{1}{1-\frac{\alpha_i}{\lambda}s}$ for $s < \lambda/\alpha_i$ (otherwise $M_i(s)$ is unbounded). The moment generating function for $\sum \alpha_i X_i$ is then given by

$$M(s) = \prod_i \frac{1}{1 - \frac{\alpha_i}{\lambda}s} \ .$$

The following upper bound holds for $M(s)$, $s < \lambda/\alpha$. An inductive argument shows that $\prod_i \frac{1}{1-\frac{\alpha_i}{\lambda}} \leq \frac{1}{1-\frac{\alpha}{\lambda}s}$ (The induction step is $\frac{1}{1-\frac{s}{\lambda}\sum_{i=1}^{k-1}\alpha_i} \cdot \frac{1}{1-\alpha_k\frac{s}{\lambda}} \leq \frac{1}{1-\frac{s}{\lambda}\sum_{i=1}^{k}\alpha_i}$). This gives $M(s) \leq \frac{1}{1-\alpha\frac{s}{\lambda}}$ for $s \leq \frac{\lambda}{\alpha}$.

Now, we can use Markov's inequality to get a bound on $\mathbf{Pr}[\sum \alpha_i X_i \geq t + \gamma]$.

$$\begin{aligned}\mathbf{Pr}[\textstyle\sum \alpha_i X_i \geq t + \gamma] &\leq \mathbf{E}\left[e^{sY}\right] \cdot e^{-s(t+\gamma)} \\ &\leq \frac{1}{1 - \frac{\alpha}{\lambda}s} \cdot e^{-s(t+\gamma)}\end{aligned}$$

Now, we choose $s = \frac{\lambda}{2\alpha} < \lambda/\alpha$. Then

$$\begin{aligned}\mathbf{Pr}[\textstyle\sum \alpha_i X_i \geq t + \gamma] &\leq 2 \cdot e^{-s(t+\gamma)} \\ &\leq e \cdot e^{-s(t+\frac{2\alpha}{\lambda})} \\ &\leq e \cdot e^{-\frac{\lambda}{2\alpha}t - 1} \\ &\leq e^{-\frac{\lambda}{2\alpha}} \\ &= \mathbf{Pr}[2\alpha Y \geq t] \ .\end{aligned}$$

■

The following lemma gives a concentration result for the sum of exponentially distributed variables.

LEMMA 6.2. *Let $X_1, X_2, \ldots, X_k \sim \text{Exp}(\lambda)$ be i.i.d. random variables. Then*

$$\mathbf{Pr}\left[\textstyle\sum_i X_i \geq 2\alpha\frac{k}{\lambda}\right] \leq e^{-(\alpha-1)k}$$

Proof. Let $M(s)$ denote the moment generating function of $\sum_i X_i$. Using standard techniques, we get

$$M(s) = \prod_i \frac{1}{1 - (s/\lambda)} = \left(\frac{1}{1 - (s/\lambda)}\right)^k$$

Using Markov's inequality gives

$$\mathbf{Pr}\left[\textstyle\sum_i X_i \geq 2\alpha\frac{k}{\lambda}\right] \leq e^{-2\alpha\frac{k}{\lambda}s}\left(\frac{1}{1 - (s/\lambda)}\right)^k$$

for all $0 < s < \lambda$. Now choosing $s = \lambda/2$ gives

$$\mathbf{Pr}\left[\textstyle\sum_i X_i \geq 2\alpha\frac{k}{\lambda}\right] \leq e^{-(\alpha-1)k}$$

■

COROLLARY 6.1. *For $\lambda = \Omega(\log n)$ the above lemma gives that*

$$\mathbf{Pr}[\textstyle\sum_i X_i \geq \alpha \cdot O(1)] \leq n^{-\Omega(\alpha)} \ ,$$

which implies that $\sum_i X_i = O(1)$ with high probability.

7 Alternative Schemes: A Brief Discussion

There were two approaches that led to improvements over the original result of Bartal [3], namely that of Bartal [4] (see also [7]), and that of Fakcharoenphol et al. [10]. While it is impossible to get a partitioning scheme with diameter Δ in which edges $e = (u, v)$ are cut with probability less than $O(\log n)\frac{d(u,v)}{\Delta}$, both of these schemes are based on the fact that clever accounting of costs can help avoid the loss of $O(\log \text{diam}(G))$ due to the recursion.

Take I: Using Seymour's Partitioning Lemma. The scheme of Bartal [4] used a graph partitioning lemma inspired by Seymour [15] that made more aggressive cuts when the resulting subgraphs were smaller (since the cost of recursion on those subgraphs would be less), and resulted in an $O(\log n \log\log n)$-distortion embedding. A similar idea was used in the subtree case by Elkin et al. [9] to get reduce their basic embedding with $O(\log^3 n)$-distortion to an improved $O(\log^2 n \log\log n)$-distortion one. It is unclear how to improve on these techniques to remove the $O(\log\log n)$ term.

Take II: Using the CKR/FRT Partitioning Lemma. The other improvement in the non-subtree

case was Fakcharoenphol et al. [10], who used a scheme in which the chance of an edge e being cut at radius Δ was approximately $\frac{d(u,v)}{\Delta} \log \frac{|B(e,2\Delta)|}{|B(e,\Delta/2)|}$. While this could be $\frac{d(u,v)}{\Delta} O(\log n)$ for some radius, the contribution nicely telescopes to give an $O(\log n)$ distortion overall. In this section, we show how to use a partitioning scheme similar to theirs to get another $O(\log^2 n \log\log n)$ distortion embedding. In fact, we only change the backward-cut subroutine in the star-decomposition:

Construct components $G_1, \ldots, G_k$ (FRT backward cuts):

Pick a random permutation π of vertices and a radius $\gamma \in [\Delta/\log n, 2\Delta/\log n]$ u.a.r.

For $i = 1, 2, \ldots$, if $v_{\pi(i)}$ is not yet assigned to any component, construct a new component as follows.

Let x_j be a portal node such that $\ell(x_j, v_{\pi(i)}) = 0$. Cut all edges at backward-edge distance γ from x_j to create a new component. (We cut edge (u,v) if $\ell(x_j,u) \le \gamma < \ell(x_j,v)$). The portal x_j becomes the root of the new component.

Note that while the vertices are considered in the order given by random permutation, the root of the components are still the portal nodes, so that we still obtain a star-decomposition. If an edge e can be cut by $v_{\pi(i)}$, then $\ell(x_j, e) \le 2\Delta/\log n$: and a necessary condition for this is that $d(v_{\pi(i)}) \le \Delta$. Thus, only the vertices in $B_d(e,\Delta)$ can centers that cut the edge e.

On the other hand, edge e is "saved" by $v_{\pi(k)}$ if the sideways distance $\ell(v_{\pi(k)}, e) < \Delta/\log n$. Thus all the vertices in $B_d(e, \Delta/\log n)$ can save the edge e. Note that edge e is cut by $v_{\pi(i)}$ if and only if $v_{\pi(i)}$ appears before all the vertices from the ball $B_d(e,\Delta/\log n)$ in the permutation π. This gives us the following claim:

CONDITION 7.1. *The probability that an edge e is cut by some backward-cut is bounded above by*

$$\frac{O(\log n)}{\Delta} \log\left(\frac{|B_d(e,\Delta)|}{|B_d(e,\Delta/\log n)|}\right).$$

Furthermore, since $\gamma \le 2\Delta/\log n$, the radius of the graph increases by only a $1+2/\log n$ factor in each level of recursion. Since the number of levels of recursion can be bounded by $O(\log n)$, we get the following result.

CONDITION 7.2. *The radius of the resulting tree is bounded above by $\Delta(1+2/\log n)^{O(\log n)} = O(\Delta)$.*

LEMMA 7.1. *The distortion of FRT based star-decomposition is bounded by $O(\log^2 n \log\log n)$.*

Proof. To bound the distortion, we sum up the contribution over different levels:

$$\begin{aligned}
&\mathbf{E}[d_T(u,v)] \\
&\quad = \sum_{\text{levels } \Delta} Pr[(u,v) \text{ is cut at level } \Delta] \cdot O(\Delta) \\
&\quad \le \sum_{\text{levels } \Delta} \frac{O(\log n)}{\Delta} \log\left(\frac{|B_d(e,\Delta)|}{|B_d(e,\Delta/\log n)|}\right) \cdot O(\Delta) \\
&\quad \le O(\log^2 n \log\log n)
\end{aligned}$$

Note that the factor $\log\log n$ appears since the sum telescopes, but in a "delayed" fashion, where each vertex appears in $O(\log\log n)$ balls around each of the edges. ∎

The reader may wonder why we work with balls $B_d(e,\Delta)$ and not with the "cones" ($B_\ell(e,\Delta)$): the reason is that the notion of cones is not stable over various levels of recursion—the vertices at a backward-edge distance r from some edge e change during the recursion in an unpredictable way, and hence it seems difficult to remove the $O(\log\log n)$ term in this analysis.

References

[1] N. ALON, R. M. KARP, D. PELEG, AND D. WEST, *A graph-theoretic game and its application to the k-server problem*, SIAM Journal on Computing, 24 (1995), pp. 78–100.

[2] B. AWERBUCH AND Y. AZAR, *Buy-at-bulk network design*, in Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS), 1997, pp. 542–547.

[3] Y. BARTAL, *Probabilistic approximations of metric spaces and its algorithmic applications*, in Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (FOCS), 1996, pp. 184–193.

[4] ———, *On approximating arbitrary metrics by tree metrics*, in Proceedings of the 30th ACM Symposium on Theory of Computing (STOC), 1998, pp. 161–168.

[5] G. CĂLINESCU, H. KARLOFF, AND Y. RABANI, *Approximation algorithms for the 0-extension problem*, in Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2001, pp. 8–16.

[6] M. CHARIKAR, C. CHEKURI, A. GOEL, AND S. GUHA, *Rounding via trees: deterministic approximation algorithms for group Steiner trees and k median*, in Proceedings of the 30th ACM Symposium on Theory of Computing (STOC), 1998, pp. 114–123.

[7] M. CHARIKAR, C. CHEKURI, A. GOEL, S. GUHA, AND S. A. PLOTKIN, *Approximating a finite metric by a small number of tree metrics*, in Proceedings of the 39th IEEE Symposium on Foundations of Computer Science (FOCS), 1998, pp. 379–388.

[8] C. CHEKURI, A. GUPTA, I. NEWMAN, Y. RABINOVICH, AND A. SINCLAIR, *Embedding k-outerplanar graphs*

into ℓ_1, in Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2003, pp. 527–536.

[9] M. Elkin, Y. Emek, D. Spielman, and S.-H. Teng, *Lower-stretch spanning trees*, in Proceedings of the 37th ACM Symposium on Theory of Computing (STOC), 2005, pp. 494–503.

[10] J. Fakcharoenphol, S. B. Rao, and K. Talwar, *A tight bound on approximating arbitrary metrics by tree metrics*, in Proceedings of the 35th ACM Symposium on Theory of Computing (STOC), 2003, pp. 448–455.

[11] N. Garg, G. Konjevod, and R. Ravi, *A polylogarithmic approximation algorithm for the group Steiner tree problem*, Journal of Algorithms, 37 (2000), pp. 66–84. Also in *9th SODA*, 1998, pages 253–259.

[12] A. Gupta, I. Newman, Y. Rabinovich, and A. Sinclair, *Cuts, trees and* ℓ_1*-embeddings of graphs*, Combinatorica, 24 (2004), pp. 233–269. Also in *Proc. 35th FOCS*, 1999, pp. 399–409.

[13] P. Indyk, *Algorithmic aspects of geometric embeddings*, in Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS), 2001, pp. 10–33.

[14] J. M. Kleinberg and É. Tardos, *Approximation algorithms for classification problems with pairwise relationships: Metric labeling and markov random fields*, Journal of the ACM, 49 (2002), pp. 616–639. Also in *Proc. 40th FOCS*, 1999, pp. 14–23.

[15] P. D. Seymour, *Packing directed circuits fractionally*, Combinatorica, 15 (1993), pp. 182–188.