Covering Approximate Shortest Paths with DAGs

Sepehr Assadi*
University of Waterloo
Canada
sepehr@assadi.info

Gary Hoppenworth†
University of Michigan
United States
garytho@umich.edu

Nicole Wein‡
University of Michigan
United States
nswein@umich.edu

Abstract

We define and study analogs of *probabilistic tree embedding* and *tree cover* for directed graphs. We define the notion of a *DAG cover* of a general directed graph G: a small collection $D_1, \ldots, D_g$ of DAGs so that for all pairs of vertices s, t, some DAG D_i provides low distortion for $\text{dist}(s,t)$; i.e. $\text{dist}_{D_i}(s,t) \leq \alpha \cdot \text{dist}_G(s,t)$, where α is the distortion.

As a trivial upper bound, there is a DAG cover with n DAGs and $\alpha = 1$ by taking the shortest-paths tree from each vertex. When each DAG is restricted to be a subgraph of G, there is a simple matching lower bound (via a directed cycle) that n DAGs are necessary, even to preserve reachability. Thus, we allow the DAGs to include a limited number of additional edges not from the original graph.

When n^2 additional edges are allowed, there is a simple upper bound of two DAGs and $\alpha = 1$. Our first result is an almost-matching lower bound that even for $n^{2-o(1)}$ additional edges, at least $n^{1-o(1)}$ DAGs are needed, even to preserve reachability. However, the story is different when the number of additional edges is $\tilde{O}(m)$, a natural setting where the sparsity of the DAG collection asymptotically matches that of the original graph. Our main upper bound is that there is a near-linear time algorithm to construct a DAG cover with $\tilde{O}(m)$ additional edges, polylogarithmic distortion, and only $O(\log n)$ DAGs. This is similar to known results for undirected graphs: the well-known FRT probabilistic tree embedding implies a tree cover where both the number of trees and the distortion are logarithmic. Our algorithm also extends to a certain probabilistic embedding guarantee. Lastly, we complement our upper bound with a lower bound showing that achieving a DAG cover with no distortion and $\tilde{O}(m)$ additional edges requires a *polynomial* number of DAGs.

CCS Concepts

• **Theory of computation** → *Sparsification and spanners*; *Random projections and metric embeddings.*

Keywords

directed graphs, graph simplification, metric embeddings

ACM Reference Format:
Sepehr Assadi, Gary Hoppenworth, and Nicole Wein. 2025. Covering Approximate Shortest Paths with DAGs. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing (STOC '25), June 23–27, 2025, Prague, Czechia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3717823.3718298

1 Introduction

Probabilistic tree embedding, first explicitly introduced by Bartal [12], and the related notion of *tree cover*, are powerful graph primitives with wide-ranging applications. The goal of a probabilistic tree embedding is to embed an (n-vertex, m-edge) undirected graph into a distribution over trees with low expected distance distortion. (Formally, for each vertex pair s, t in the original graph G, (1) every tree T in the distribution $\mathcal{D}$ satisfies $\text{dist}_G(s,t) \leq \text{dist}_T(s,t)$, and (2) $\mathbb{E}_{T\sim\mathcal{D}}[\text{dist}_T(s,t)] \leq \alpha \cdot \text{dist}_G(s,t)$ where α is the *distortion*.) Tree covers provide a somewhat more relaxed guarantee wherein the goal is to obtain a small collection of trees so that for each pair u, v of vertices, *some* tree in the collection provides low distortion for $\text{dist}(s,t)$.

Famously, building upon prior work [3, 12, 13, 53], Fakcharoenphol, Rao, and Talwar [39] obtained the asymptotically optimal bound of $O(\log n)$ distortion for probabilistic tree embeddings. Fast computation of this construction has been extensively studied [22, 62], as well as extensions to various settings such as dynamic [42], online [12, 15], distributed [45, 56], parallel [23, 43], hop-constrained [47], and derandomized [33]. These constructions have enjoyed a strikingly varied set of applications such as k-median, buy-at-bulk network design [23], generalized Steiner forest, minimum routing cost spanning tree, multi-source shortest paths [56], distance oracles [22], linear systems [36], minimum bisection [54] oblivious routing [48, 63, 64], metric labeling [57], and group Steiner tree [44].

A probabilistic tree embedding implies a tree cover with the same distortion and $O(\log n)$ trees, by sampling trees from the distribution. For tree cover there are also additional trade-offs between distortion and number of trees [8, 9, 14, 16, 65]: for any integer $k \geq 1$, one can achieve distortion $2k-1$ with $O(n^{1/k}\log^{1-1/k} n)$ trees. This result has applications, for instance, to routing [7–9] and distributed directories [10]. Tree covers have also been fruitful for metrics with geometric structure such as Euclidean space, doubling metrics, ultrametrics, and planar graphs [5, 16, 17, 31, 32, 61]. Applications in these settings include, for instance, routing [46, 52], spanners [5, 40, 52], and distance labeling [46].

While both probabilistic tree embeddings and tree covers have experienced wide adoption, the success of these techniques has been limited to *undirected* graphs. The contribution of the current

*This work was initiated while the author was at Rutgers University. Sepehr Assadi is supported in part by a Sloan Research Fellowship, an NSERC Discovery Grant (RGPIN-2024-04290) and a Faculty of Math Research Chair grant.
†This work was supported by NSF:AF 2153680.
‡This work was initiated while the author was at Rutgers University supported by a grant to DIMACS from the Simons Foundation (820931), and continued while the author was at the Simons Institute.

STOC '25, Prague, Czechia

ACM ISBN 979-8-4007-1510-5/25/06
https://doi.org/10.1145/3717823.3718298

work is to define and prove bounds for *directed analogs* of these primitives.

Before elaborating on the details, we discuss the wider context of this work.

Graph Simplification for Distance Preservation. This work falls under the umbrella of *graph simplification* for distance preservation, where the goal is to obtain a simplified representation of a graph G while (approximately) preserving its distances. In addition to probabilistic tree embeddings and tree covers, a number of other such structures exist, including spanners, emulators, distance preservers, hopsets, and distance oracles (see e.g. the survey [1]). However, distances in directed graphs suffer from a relatively sparse toolkit of structural primitives. Each of the above structures can be defined for both undirected and directed graphs, however they are either provably nonexistent for directed graphs (e.g. spanners, distance oracles), or generally less-understood (e.g. distance preservers, hopsets). For hopsets on directed graphs, there has been recent significant progress [20, 26, 58, 66], though there still remain polynomial gaps between upper and lower bounds (in contrast to undirected graphs [38, 50]).

Furthermore, this lack of structural primitives for directed graphs is perhaps a contributing factor to our gaps in understanding of basic algorithmic problems regarding distances and reachability in directed graphs. As a few examples, there are large gaps in our understanding of even the single-source reachability problem in various common settings such as streaming, distributed, and parallel; in addition, the following basic problems are less-understood for directed graphs than undirected: diameter approximation [30], disjoint shortest paths [60], not-shortest path [59]. See Section 1.3 of [2] for more detail. The current work can be viewed as a step towards remedying the lack of structural primitives for distances in directed graphs.

DAGs. Our goal is to define a directed analog of probabilistic tree embedding and tree cover. We will use directed acyclic graphs (DAGs) as our directed analog of trees. This is a natural choice, not only because DAGs are among the most natural directed analogs of trees[1], but also because many problems are much easier on DAGs than general directed graphs. This is important because a central purpose of graph simplification for distance preservation is to get better algorithms for distance-related problems, by first simplifying the graph, and then applying an algorithm to the simplified graph. This approach only works when the simplified graph indeed allows for more efficient algorithms.

There are many example of distance-related problems that are much easier on DAGs than general directed graphs, but we will specify a few. Single-source shortest paths with negative weights in DAGs in linear time is an undergraduate exercise, while for general directed graphs it is a notorious problem that has witnessed several recent breakthroughs [19, 27, 41, 51], and remains open. For distance preservers, the best known results on DAGs are better than those for general directed graphs [24]. For hopsets in DAGs, upper bounds for the simpler problem of shortcut sets tend to extend quite easily, whereas this is not the case for general directed graphs [20, 58].

Defining the problem. For simplicity, we will focus our initial discussion on directed analogs of tree cover (rather than probabilistic tree embedding). Our goal is to obtain a small collection of DAGs such that for each ordered pair s, t of vertices, some DAG in the collection provides low distortion for $\text{dist}(s, t)$.

Our first observation is that there is a trivial upper bound of n DAGs with no distortion, because one can always return the collection of shortest paths DAGs, one from each of the n vertices. Ultimately, we would like to do much better, and achieve a bound similar to what [39] yields: polylogarithmic bounds on both the distortion and the number of DAGs.

Suppose we require each DAG in the collection to be a subgraph of the original graph. Then, a simple construction shows that no non-trivial upper bound is possible: Consider a directed cycle. Suppose s and t are adjacent vertices where s appears right after t on the cycle. Then, the only way to preserve the reachability from s to t is to include the unique path from s to t as a DAG in the collection; note that this path includes every edge in the cycle except (t, s). Thus, to have finite distortion, the DAG collection must include such a path for all n pairs s, t of adjacent vertices on the cycle.

Thus, we need to relax the problem to allow the DAGs to contain extra edges not in the original graph. If we allow the DAGs to have arbitrarily many edges, this trivializes the problem: There is a simple upper bound achieving only two DAGs and no distortion: Consider an arbitrary ordering of the vertices. Construct one DAG by including all $\binom{n}{2}$ forward edges with respect to the ordering, where the weight of each edge (u, v) is the distance $\text{dist}(u, v)$ in the original graph. Construct the other DAG in the same way, but for the opposite ordering of vertices. This way, for each pair of vertices s, t, at least one of the two DAGs witnesses no distortion of the distance $\text{dist}(s, t)$. Beyond trivializing the problem, allowing arbitrarily many extra edges complicates the graph, which is the opposite of our goal of graph simplification.

Thus, we allow the DAGs to include a *limited* number of edges not from the original graph. The number of additional edges can be expressed in terms of n and/or m. Now, we have arrived at the main definition of the problem:

Definition 1.1 (DAG cover) Let G be an n-node, m-edge weighted, directed graph. We define a DAG cover of G with $f = f(m, n)$ additional edges and $g = g(m, n)$ DAGs as a collection $D_1, \ldots, D_g$ of DAGs such that

- each DAG D_i is defined over the vertex set $V(G)$ of G, and
- the collection of DAGs contains at most additional f edges not in $E(G)$, i.e., $|(\cup_i E(D_i)) \setminus E(G)| \leq f$.

We say that a DAG cover $D_1, \ldots, D_g$ of G is *reachability-preserving* if for all nodes $s, t \in V(G)$, s can reach t in G if and only if there exists a DAG D_i such that s can reach t in D_i.

We say a DAG cover $D_1, \ldots, D_g$ of G is *α-distance-preserving* if for all nodes $s, t \in V(G)$,

$$\text{dist}_G(s, t) \leq \min_{i \in [g]} \text{dist}_{D_i}(s, t) \leq \alpha \cdot \text{dist}_G(s, t).$$

[1] Another natural directed analog of a tree is an arborescence, which is not meaningful in this context because a trivial lower bound shows that for a directed complete bipartite graph $A \to B$, one would need n arborescences to preserve even the reachability for all pairs of vertices.

Our Results. Our first result is essentially the strongest possible impossibility result for the regime where the number of additional edges is in terms of n: Even reachability-preserving DAG covers with almost n^2 additional edges require almost n DAGs (where n DAGs is a trivial upper bound, as previously mentioned):

Theorem 1.2 *There exists a family of n-node directed graphs G such that any reachability-preserving DAG cover of G with at most $n^{2-o(1)}$ additional edges requires at least $n^{1-o(1)}$ DAGs.*

An implication of Theorem 1.2 is to rule out a certain approach for obtaining hopsets from shortcut sets. [58] obtained a breakthrough shortcut set construction and extended it to a hopset construction for DAGs with the same bounds; they also extended it to general directed graphs, but with worse bounds. Subsequently, [20] obtained an involved algorithm yielding a hopset for general directed graphs with bounds matching the original bound of [58]. If there existed a $(1+\varepsilon)$-distance-preserving DAG cover with $\tilde{O}(n)$ additional edges and a polylogarithmic number of DAGs, then one could skip the involved algorithm of [20], and obtain this result as an immediate corollary of [58]. However, Theorem 1.2 implies that such an approach is too good to be true.

Theorem 1.2 effectively rules out almost any non-trivial construction when the number of additional edges is in terms of n. However, as an aside, we show that the term "almost" here is crucial; there is indeed a non-trivial construction when the number of edges is slightly less than n^2 additional edges:

Theorem 1.3 *Every n-node directed graph G admits an exact distance-preserving DAG cover of G with $O\left(\frac{n^2(\log\log n)^4}{\log^2 n}\right)$ additional edges and $n^{o(1)}$ DAGs.*

Ultimately, the achievable bounds are unsatisfactory when the number of added edges is expressed in terms of n, so we turn our attention to expressing this quantity in terms of m. In particular, in line with our goal of graph simplification, we pay special attention to the natural regime of $\tilde{O}(m)$ additional edges, where the sparsity of the DAG collection is asymptotically the same as that of the original graph. Perhaps surprisingly, this distinction between n and m is crucial for obtaining better bounds. We see a separation between the n and m regimes by observing that every graph G admits a reachability-preserving DAG cover with $O(m)$ additional edges and only two DAGs.

Observation 1.4 *Every graph G admits a reachability-preserving DAG cover with $O(m)$ additional edges and only two DAGs.*

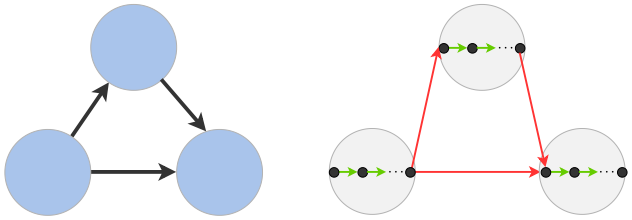

Figure 1: On the left is a directed graph G, with strongly connected components shown in blue. On the right is one of the two DAGs constructed in our reachability-preserving DAG cover of G in Observation 1.4. The other DAG in our cover can be obtained by reversing all the green edges in the diagram.

Proof sketch of Observation 1.4. Let $[1,k]$ denote the set of strongly connected components (SCCs) in graph G. For each SCC i of G, pick an arbitrary ordering of its vertices, and let f_i and ℓ_i be the first and last vertices in this ordering. Define one of the two DAGs as a directed path through each SCC according to its ordering, unioned with the following edges: for each SCC i, (1) an edge from ℓ_i to f_j for all j such that G has an edge from SCC i to SCC j, and (2) an edge to f_i from ℓ_j for all j such that G has an edge from SCC j to SCC i. Define the other DAG as a directed path through each SCC according to the reverse of its ordering. This is a reachability-preserving DAG cover because the first DAG preserves reachability for vertices in different SCCs, while for vertices in the same SCC, one of the two DAGs preserves reachability depending on which vertex comes first in the SCC's ordering. □

Thus, the interesting regime for $\tilde{O}(m)$ additional edges is (approximate) distance-preserving DAG covers.

Question: How many DAGs are required for α-distance-preserving DAG covers with $\tilde{O}(m)$ additional edges? For which values of α is it polynomial versus polylogarithmic?

We make progress on this question from both the upper and lower bound fronts: the answer is *logarithmic* for certain polylogarithmic values of α, and the answer is *polynomial* for exact distances ($\alpha = 1$). Our bounds are stated in Theorems 1.5 and 1.6.

Our upper bound in Theorem 1.5 achieves similar bounds to what [39] yields for undirected graphs: polylogarithmic bounds on both the distortion and the number of DAGs. Furthermore, the fact that only $\tilde{O}(m)$ edges are added means that the sparsity of the DAG collection is asymptotically the same as that of the original graph, which satisfies our original goal of graph simplification. Additionally, such a DAG cover can be computed in near-linear time.

Theorem 1.5 *Let G be an n-node weighted, directed graph with positive integer edge weights and polynomial aspect ratio. Then there exists an $O(\log^4 n)$-approximate DAG cover with $O((m+n)\log^3 n)$ additional edges and $O(\log n)$ DAGs. Moreover, this DAG cover can be computed w.h.p. in $O(m\log^4 n + n\log^5 n)$ time.*

To complement our upper bound, and to show a separation between reachability versus exact shortest paths for $\tilde{O}(m)$ additional

edges, we obtain the following polynomial lower bound on the number of DAGs.

Theorem 1.6 *There exists a family of n-node, m-edge directed graphs G with positive integer edge weights and polynomial aspect ratio such that any exact distance-preserving DAG cover of G with $m^{1+\varepsilon}$ additional edges requires $\Omega(n^{1/6})$ DAGs, for a sufficiently small constant $\varepsilon > 0$.*

So far we have focused on DAG covers as an analog to tree covers, but there is also a corollary of Theorem 1.5 that analogizes probabilistic tree embedding. Recall that the guarantee of a probabilistic tree embedding is that for each pair of vertices, the expected distortion of their distance over the distribution of trees, is bounded. It is impossible to achieve a precisely analogous guarantee for a distribution over DAGs because in any DAG, for every pair of vertices s, t, either $\text{dist}(s,t)$ or $\text{dist}(t,s)$ is infinite. Instead, we define a distribution $\mathcal{D}$ over DAGs so that for all pairs of nodes (s,t) in the transitive closure of G, if a DAG is chosen from $\mathcal{D}$, the reachability from s to t is preserved with probability $1/2$, and conditioned on that, the expected distortion is polylogarithmic:

Theorem 1.7 *Let G be an n-node weighted, directed graph with positive integer edge weights and polynomial aspect ratio. Then there exists a distribution $\mathcal{D}$ of DAGs over vertex set $V(G)$ satisfying the following properties:*

1. *The support of distribution $\mathcal{D}$ contains at most $O((m+n)\log^3 n)$ additional edges, and*
2. *Let $D \sim \mathcal{D}$ be a DAG sampled from distribution $\mathcal{D}$. For all pairs of nodes $(s,t) \in TC(G)$, s can reach t in D with probability $1/2$. Moreover,*

$$\mathbb{E}[\text{dist}_D(s,t) \mid s \rightsquigarrow_D t] = O(\log^4 n \cdot \text{dist}_G(s,t)),$$

where $s \rightsquigarrow_D t$ denotes the event that s can reach t in D.

Finally, we return to the case of DAG covers with *no* additional edges, for graphs with bounded diameter. We note that the above trivial lower bound of n DAGs does not apply for graphs of low diameter, since the construction is simply a cycle of diameter $n-1$. In fact, there is a simple upper bound for low diameter graphs: an exactly distance-preserving DAG cover whose number of DAGs is logarithmic in n but factorial in the diameter D. To see this, randomly order the vertices in G and construct a DAG that includes every edge in G that goes "forward" with respect to the ordering. A given shortest path is included in this DAG if its (at most $D+1$) vertices are ordered consistently with the random ordering, which occurs with probability $\geq 1/(D+1)!$. Then, the number of random trials (and thus number of DAGs) needed to include all $\Theta(n^2)$ shortest paths in some DAG with high probability is $O((D+1)! \cdot \log n)$.

Is exponential dependence on D necessary? For our final result, we show that it is:

Theorem 1.8 *There exists a family of n-node directed graphs G with diameter $O(\log n)$ such that any reachability-preserving DAG cover of G with no additional edges requires n DAGs.*

Additional Related Work. A recently developing area of research focuses on understanding the *structure* of shortest paths in both directed and undirected graphs [2, 4, 11, 18, 25, 34, 35, 37]. See Section 1.4 of [18] for more details. Our work can be viewed as a contribution to this undertaking.

Additionally, the following papers are related to the wider context of our work: embedding directed planar graphs into directed ℓ_1 [55], an extension of treewidth to DAG-width [21], and shortest-path preservers with minimum aspect ratio [18].

2 Technical Overview

2.1 Lower Bounds

First we will outline some of the ideas behind Theorem 1.2 (our lower bound that any reachability-preserving DAG cover with almost n^2 additional edges requires almost n DAGs), and then we will turn to Theorem 1.6 (our lower bound that any distance-preserving DAG cover with $m^{1+\varepsilon}$ additional edges requires $\Omega(n^{1/6})$ DAGs), which builds upon these ideas.

Number of additional edges in terms of n. Our starting point is a family of graphs whose variants are commonly used for obtaining lower bounds for related structures including hopsets/shortcut sets, spanners/emulators, and reachability/distance preservers, first used by [49]. Each graph in this family is a DAG that contains a *large* collection of *long* paths that are pairwise *edge-disjoint*, and each path is the *unique* path between its endpoints.

Of course, such a graph does not yield a lower bound for DAG covers because the graph itself is already a DAG. Instead, the intuition is that we can modify the above "base" graph so that a constant fraction of pairs of intersecting paths are "incompatible", i.e., cannot appear together in the same DAG. To do so, we replace every vertex v in the base graph with two vertices v_1, v_2 that have a bidirectional edge between them. Then, we randomize which of these two vertices constitute the entry and exit points of each path passing through v. That is, for every edge in the base graph that enters (respectively, exits) v, we assign it to enter (respectively, exit) either v_1 or v_2 with probability $1/2$ each. Then, for any pair of paths that intersect at v, the probability that they traverse the edge (v_1, v_2) in opposite directions is $1/4$. If this happens, the union of these two paths contains a cycle, making them incompatible. A probabilistic analysis of all intersections between paths yields a lower bound on the number of DAGs needed to accommodate every path.

We are not done, however, because there are added edges. In particular, these added edges could bypass the (v_1, v_2) bidirectional edge, allowing two paths that intersect at v to become compatible. A key property of the base graph is that each pair of paths intersects on at most one vertex. This means that any added edge can only "help" one path. Furthermore, the parameters of the base graph are set so that the number of paths is a constant factor larger than the number of added edges. This means that a constant fraction of the paths are not helped by any added edge. Then, we can again perform a probabilistic analysis to calculate a bound on the number of DAGs needed to accommodate all of the non-helped paths. To obtain this bound, it is important that the paths are long enough (a property of the base graph) because for a DAG to accommodate a path, the path needs to randomly chose entry/exit points so that it is consistent with the DAG at *every* step.

This analysis allows us to obtain essentially the strongest possible impossibility result for the regime where the number of additional edges is in terms of n: reachability-preserving DAG covers with almost n^2 additional edges, require almost n DAGs.

Number of additional edges in terms of m. One issue towards obtaining a lower bound for $m^{1+\varepsilon}$ added edges is the previously mentioned upper bound of a reachability-preserving DAG cover with two DAGs. That is, in contrast to the case of $n^{2-o(1)}$ added edges, any lower bound for $m^{1+\varepsilon}$ added edges must hold only for exact or approximate shortest paths, and not for reachability.

A related issue is that if the SCCs are of polylogarithmic size, there is an upper bound for exact-distance-preserving DAG covers: For each SCC S and each vertex $v \in S$, add (appropriately weighted) edges from v to the union of the out-neighbors of S, and add edges to v from the union of in-neighbors of S. This DAG handles vertex pairs in two different SCCs, and a polylogarithmic number of additional DAGs handle vertex pairs in the same SCC.

Therefore, to achieve a lower bound, the SCCs must be super-polylogarithmic in size. Specifically, each of our SCCs will be a clique of size $c = n^\delta$ with bidirectional edges. Similar to our construction for $n^{2-o(1)}$ added edges, every vertex in the original graph is replaced with a clique, and every incoming and outgoing edge to/from the clique picks a *random* clique vertex to enter or exit, respectively. Now, a pair of paths is only incompatible if each traverses the same clique edge in opposite directions. This occurs much more rarely than in our previous construction with cliques of size 2, but we show that this construction can still be used towards getting a polynomial lower bound on the number of DAGs.

However, there is a major issue with the construction so far: in the previous construction it was important for the number of paths to be larger than the number of added edges (since each added edge could only "help" one path). However, since the paths in the base graph are edge-disjoint, the number of paths is necessarily less than the number $m^{1+\varepsilon}$ of added edges. For this reason, we need to relax the edge-disjointness condition. This has been done in prior work (first by [49]) for other problems (e.g. for shortcut sets with m added edges), by taking a certain type of *product* of two copies of the base graph. We use such a product graph, which allows pairs of paths to overlap on a single edge (rather than just a vertex). Using this product graph as our new base graph, we perform the previously described clique-replacement step. However, the use of a product graph significantly complicates the proof for two reasons:

(1) Since pairs of paths can overlap on an edge in the product graph, there are two types of added edges: "long" ones that only help one path as before, and "short" ones between adjacent cliques, which could help *many* paths that all share an edge between those two cliques.

(2) In our previous construction (for $n^{2-o(1)}$ added edges), we used the fact that each path picks its entry/exit points (v_1 or v_2) *independently*. This is no longer the case. These choices are made independently for each edge, but since pairs of paths can share an edge, there are correlations between whether or not paths are consistent with a fixed DAG.

To circumvent this correlation issue, we restrict our attention to *edge-disjoint* sets of paths going through the same clique. For such paths, their entry-exit points to/from the clique are chosen independently and we use this to calculate their probability of compatibility. Such a set of paths going through a given clique can be viewed as a *matching* in an appropriate auxiliary graph. We are interested in lower bounding the size of such matchings over all of the cliques (see the full version of this paper for details).

We can think of this scenario as a game played with an adversary: First we randomly choose the trajectories of all of the paths through the cliques. Then the adversary, after seeing the random choices, is allowed to help paths using "long" edges, and help *parts* of paths using "short" edges. The goal of the adversary is to minimize the matching sizes in the non-helped portions of the graph.

Due to the intricacy of this scenario, to argue about the structure of the non-helped parts of the graph, we need to open the black box of the construction of the base graph and product graph, which are based on constructions from discrete geometry. In particular, we need to prove a special "expansion" property of these graphs.

2.2 Upper Bound

In this section we will describe some of the ideas behind Theorem 1.5, our $O(\log^4 n)$-approximate DAG cover with $\tilde{O}(m)$ additional edges and $O(\log n)$ DAGs. Our starting point is the technique of [39] for probabilistic tree embedding in undirected graphs, which is the elegant and well-known technique of a recursively applying a *low-diameter decomposition* (LDD). In a low-diameter decomposition, we delete edges from the input graph according to a random process, so that afterwards each resulting connected component has bounded diameter, and the probability that any given path survives has a particular inverse relation to its length. Then to build a tree, we can add edges to connect these connected components into a star, where each edge weight is the diameter of the original graph (so that no distances are shortened), and we can recurse on each connected component.

LDDs for directed graphs are also known [19, 28, 29]; in particular, we use Lemma 1.2 of [19]. Instead of each connected component having bounded diameter, each *strongly* connected component (SCC) has bounded diameter. The other guarantee is similar to the undirected case: the probability that any given path survives has a particular inverse relation to its length. We recursively apply this LDD to each SCC. While this process works smoothly, the issue is that it is not clear how to use this recursive LDD to obtain a DAG cover. We highlight a few considerations:

(1) In the undirected case, the above random process results in a single tree. In contrast, in any DAG at least half of all ordered vertex pairs have infinite distance, so we need to construct multiple DAGs. To do so, we need to establish a partition of the ordered vertex pairs to determine which distances will be preserved in each DAG we construct, and which distances will be neglected.

(2) Consider a vertex pair (s, t) and a DAG that approximately preserves $\text{dist}(s, t)$. The topological order of the DAG may not respect the order of vertices a given st-path, which could include many segments that are "backwards" with respect to the topological order of the DAG. To resolve this issue, we need to add a limited number of edges to *bypass* these backwards segments for all such s, t.

(3) If we were to recurse on multiple SCCs to build our DAG collection, each SCC would return multiple DAGs (otherwise some

distances would be infinite). Then, we would need to combine these DAGs in such a way that globally all distances are preserved (approximately). If we consider all possible ways to combine these DAGs, we risk a combinatorial explosion of combinations, leading to a large number of DAGs. To address this issue, instead of using recursion as a black box, we take a more global approach and define the process so that we construct a total of only two DAGs. (And then, similar to the undirected version, we repeat the random process $O(\log n)$ times to obtain the final collection of DAGs).

The rough idea of our algorithm is as follows. Consider a pair of vertices (s, t), and consider the moment during the construction of the recursive LDD at which an edge is deleted, causing $\text{dist}(s, t)$ to become infinite. There are two cases: (1) right before this moment, s and t were in the same SCC, or (2) they were in different SCCs. In case 1, if s and t were in the same SCC S, then the LDD gives us a bound on $\text{dist}(s, t)$. Similar to the undirected version, we would like to add edges of weight $\text{diameter}(S)$ to create a star-like structure to approximately preserve $\text{dist}(s, t)$. To make a star-like DAG for each SCC, we use a folklore shortcut set construction, which creates a DAG of diameter 2 with $\widetilde{O}(|S|)$ added edges.

In case 2, if s and t were in different SCCs, this means that s appears before t in the topological order of the evolving graph resulting from the recursive LDD. However, some edge on an st-path was deleted, disconnecting t from s. We would like to bypass this edge, by adding a limited number of new edges that are consistent with the topological order emerging from the recursive LDD. We do this roughly via another star-like DAG for each SCC, as well as adding additional edges to link the SCCs together. This linking of the SCCs is done in a roughly similar way to the previously described reachability-preserving DAG cover with $\tilde{O}(m)$ added edges. This is the part of the construction that requires $\tilde{O}(m)$ added edges, as opposed to $f(n)$ edges.

To approximately preserve all distances, we execute both of the above cases for every SCC and every level of the LDD. Importantly, we cannot construct a separate DAG for every level of the LDD because we need a low-distortion path from s to t to be contained in *a single DAG*. That is, if an st-path requires multiple "bypass" edges from multiple levels of the LDD then all of these bypass edges need to appear in the *same* DAG. A key challenge is to ensure that the union of *all* of the added edges over all levels genuinely forms only two DAGs, and no cycles are formed from interference between edges from different levels.

3 Paper Structure

In the remainder of the paper, we present our construction of $\widetilde{O}(1)$-approximate DAG covers with $\widetilde{O}(m)$ additional edges in Theorem 1.5. Additionally, we give an abbreviated proof that this DAG cover satisfies the properties claimed in Theorem 1.5. We defer the full proof of this theorem and the remaining theorems referenced in Section 1 to the arXiv version of our paper [6].

4 An Upper Bound for DAG Covers with $\widetilde{O}(m)$ additional Edges

The goal of this section is to prove the following theorem.

Theorem 1.5 *Let G be an n-node weighted, directed graph with positive integer edge weights and polynomial aspect ratio. Then there exists an $O(\log^4 n)$-approximate DAG cover with $O((m+n)\log^3 n)$ additional edges and $O(\log n)$ DAGs. Moreover, this DAG cover can be computed w.h.p. in $O(m \log^4 n + n \log^5 n)$ time.*

An essential tool we will need to prove Theorem 1.5 is the directed low-diameter decomposition due to [19]. The key properties of this low-diameter decomposition are summarized in Lemma 1.2 of [19], which we restate below.

Lemma 4.1 (Lemma 1.2 of [19]) *Let $G = (V, E, w)$ be an n-node weighted, directed graph with positive integer edge weights. Let d be a positive integer such that $d = O(\text{poly}(n))$. Then there exists a set of edges $E' \subseteq E$ with the following properties:*

(1) each SCC of $G \setminus E$ has weak diameter at most d in G, and
(2) for every edge $e \in E$,

$$\Pr[e \in E'] = O\left(\frac{w(e) \cdot \log^2 n}{d}\right).$$

The set E' and the SCCs of $G \setminus E'$ can be computed in $O(m \log^2 n + n \log^3 n)$ time.

4.1 Construction of DAG Cover $\mathcal{D}$

Let G be an n-node, weighted, directed graph with positive integer edge weights and maximum weight W. The key step of our construction will be a hierarchical decomposition of G obtained by repeatedly applying Lemma 4.1. Let $G_0 = G$, and let $\mathcal{F}_0$ denote the collection of all strongly connected components in G_0. We will construct set family $\mathcal{F}_{i+1}$ and graph G_{i+1} from graph G_i, as follows.

(1) Apply Lemma 4.1 to graph G_i with integer parameter $d_i = n^2 W \cdot 2^{-i-1}$. Let $E_i \subseteq E(G_i)$ be the set of edges specified by Lemma 4.1.
(2) Let $G_{i+1} = G_i - E_i$, and let $S^1_{i+1}, \ldots, S^{k_{i+1}}_{i+1} \subseteq V(G)$ denote the SCCs of G_{i+1}. We define $\mathcal{F}_{i+1} \subseteq 2^{V(G)}$ to be the set family
$$\mathcal{F}_{i+1} = \bigcup_{j \in [1, k_{i+1}]} \{S^j_{i+1}\}.$$
(3) Let $z = \lceil \lg(n^2 W) \rceil$. We will terminate our recursion after computing graph G_z and collection $\mathcal{F}_z$.

Note that since $d_z < 1$, the graph G_z will be a DAG. We will use D^* to denote graph G_z. The following claim summarizes the properties of the sets $\mathcal{F}_i$ inherited from Lemma 4.1.

Claim 4.2 *The following properties of the $\mathcal{F}_i$'s and E_i's are inherited from Lemma 4.1:*

(1) Set family $\mathcal{F}_i$ is the collection of SCCs of graph G_i.
(2) Each set $S \in \mathcal{F}_i$ has weak diameter at most d_i in G_{i-1} (and consequently, in G as well).
(3) For every edge $e \in E(G_i)$,

$$\Pr[e \in E_i] = O\left(\frac{w(e) \cdot \log^2 n}{d_i}\right).$$

We will need several preliminary notations and claims before we can construct the DAGs associated with set family of $\mathcal{F}_i$'s. Fix an $i \in [0, z]$. We define a total order $<_i$ on the SCCs $\mathcal{F}_i$ in G_i. Informally, this will correspond to a specific topological order of

the condensation graph of G_i. Fix two SCCs $S, T \in \mathcal{F}_i$, where $S \neq T$. If there exists a node $s \in S$ and a node $t \in T$ such that s can reach t in graph G_i, then we let $S <_i T$. If there exists distinct sets $S', T' \in \mathcal{F}_{i-1}$ satisfying $S \subseteq S'$ and $T \subseteq T'$ such that $S' <_{i-1} T'$, then we let $S <_i T$, as well. We observe that if there exists distinct $S, T \in \mathcal{F}_i$ such that $S <_i T$ and $T <_i S$, then that implies a contradiction of Property 1 of Claim 4.2. We conclude that $<_i$ is a partial order on the elements of $\mathcal{F}_i$. Finally, we convert partial order $<_i$ to a total order by completing the ordering arbitrarily.

Claim 4.3 *Relation $<_i$ is a total ordering of set family $\mathcal{F}_i$.*

We will now define a total order $<_D$ on the vertex set $V(G)$ that respects every total order $<_i$ for all $i \in [0, z]$. For each node $v \in V(G)$ and each $i \in [0, z]$, we let $S_i(v) \in \mathcal{F}_i$ denote the unique set in $\mathcal{F}_i$ containing node v. Given $s, t \in V(G)$, we will let $s <_D t$ if

$$s \neq t \qquad \text{and} \qquad S_i(s) <_i S_i(t),$$

for some $i \in [0, z]$. We will let $<_D$ be an arbitrary total order that satisfies this property for all $s \neq t \in V(G)$. We will use $s \leq_D t$ to denote that either $s <_D t$ or $s = t$.

Claim 4.4 *Relation $<_D$ is a total ordering of vertex set $V(G)$.*

Proof. Suppose towards contradiction that there exist nodes $s, t \in V(G)$, with $s \neq t$, such that $s <_D t$ and $t <_D s$. Then without loss of generality, $S_i(s) <_i S_i(t)$ and $S_j(t) <_j S_j(s)$ for some $i \leq j$. Since $S_j(s) \subseteq S_i(s)$ and $S_j(t) \subseteq S_i(t)$ and $S_i(s) <_i S_i(t)$, we conclude that $S_j(s) <_j S_j(t)$ by the definition of $<_j$. This contradicts Claim 4.3, so we conclude that $<_D$ is a total ordering of $V(G)$. □

For each set $S \in \mathcal{F}_i$, we associate two *representative* nodes $r_1(S), r_2(S) \in S$. We choose $r_1(S)$ to be the first node in S under total order $<_D$, and we choose $r_2(S)$ to be the last node in S under total order $<_D$. For each node $v \in V(G)$ and each $i \in [0, z]$, we define the level i representative nodes of v, denoted as $r_i^1(v)$ and $r_i^2(v)$, to be

$$r_i^1(v) = r_1(S_i(v)) \quad \text{and} \quad r_i^2(v) = r_2(S_i(v)).$$

In particular, when $i = z$, we have that $r_z^1(v) = r_z^2(v) = v$, since D^* is a DAG. For each edge $(u, v) \in E(G)$, we define the *level* of edge (u, v), denoted as $\ell(u, v)$, to be the smallest integer $i \in [0, z]$ such that $S_i(u) \neq S_i(v)$.

In addition to the above notation, we will need the following claim, which roughly states that we can approximately preserve distances between nodes s, t in an SCC $S \in \mathcal{F}_i$ when $s <_D t$, using a small collection of edges H that respect total order $<_D$.

Claim 4.5 *Let $S \in \mathcal{F}_i$ be a set in family $\mathcal{F}_i$. Then there exists a set of directed, weighted edges $H \subseteq S \times S$ of size $|H| = O(|S| \log |S|)$ with the following properties:*

(1) Every edge (u, v) in H respects total order $<_D$.
(2) Every edge (u, v) in H has weight $w(u, v) = d_i$.
(3) For all $s, t \in S$ with $s <_D t$,

$$\text{dist}_G(s, t) \leq \text{dist}_H(s, t) \leq 2d_i.$$

(4) Set H can be computed in $O(|H|)$ time.

Proof. Let $S = \{s_1, \ldots, s_k\}$ be the nodes in S, ordered with respect to $<_D$. We define a path π that respects $<_D$ as follows:

$$\pi = (s_1, \ldots, s_k).$$

We assign each edge in π weight d_i. Initially, we let $H = E(\pi)$. By [58], for a directed path π on k nodes, there exists a collection of directed edges H' such that

- $|H'| = O(k \log k)$ and H' can be computed in $O(|H'|)$ time,
- H' is contained in the transitive closure of path π, and
- For every pair of nodes s, t in path π such that node s occurs on π before node t, there exists an $s \rightsquigarrow t$ path π' in $\pi \cup H'$ with $|\pi'| \leq 2$ edges.

Then we let $H = E(\pi) \cup H'$, with every edge in H assigned weight d_i. Properties 1, 2, and 4 of our claim are clearly satisfied. We now verify Property 3 for all $s, t \in S$ with $s <_D t$:

- By Property 2, $\text{dist}_H(s, t) \geq d_i$. Moreover, $d_i \geq \text{dist}_G(s, t)$ by Claim 4.2, so $\text{dist}_G(s, t) \leq \text{dist}_H(s, t)$.
- For each pair of nodes s, t in path π such that $s <_D t$, there exists an $s \rightsquigarrow t$ path π' in H with $|\pi'| \leq 2$ edges. Then $\text{dist}_H(s, t) \leq w(\pi') \leq 2d_i$, by Property 2. □

We construct two DAGs D_1 and D_2 associated with our collection of $\mathcal{F}_i$'s. DAG D_1 will respect total order $<_D$, while DAG D_2 will respect the reverse of $<_D$. We construct DAG D_1 as follows.

Construction of DAG D_1.

(1) The vertex set of D_1 will be the same as G, i.e., $V(D_1) = V(G)$.
(2) For each edge $(u, v) \in E(D^*)$, each $i, j \in [0, z]$ such that $\min(i, j) \geq \ell(u, v)$, and each $k, k' \in \{1, 2\}$, add the edge $(r_i^k(u), r_j^{k'}(v))$ to D_1. Assign edge $(r_i^k(u), r_j^{k'}(v))$ weight $w_G(u, v) + d_i + d_j$. Since $r_z^1(v) = r_z^2(v) = v$ for all $v \in V(G)$, this procedure also adds edges of the form $(u, r_i^j(v))$ and $(r_i^j(u), v)$, for all $i \geq \ell(u, v)$ and $j \in \{1, 2\}$. Additionally, we add the edges in $E(D^*)$ with their original weights in G to DAG D_1.
(3) Fix an index $i \in [0, z]$, and fix a set $S_i^j \in \mathcal{F}_i$, where $j \in [1, k_i]$. Let $H_i^j \subseteq S_i^j \times S_i^j$ be the set of directed, weighted edges specified in Claim 4.5 with respect to set S_i^j. We will add the set of edges

$$H = \bigcup_{i \in [0,z], j \in [1,k_i]} H_i^j$$

to D_1.

This completes the construction of D_1. If there are parallel edges in D_1 from node u to node v, then we keep only the lowest weight edge from u to v. We will quickly verify that D_1 is a DAG.

Claim 4.6 *Graph D_1 is a DAG.*

Proof. Note that every edge in H respects total order $<_D$ by Claim 4.5. We will now show that every edge we add to D_1 in Step 2 respects $<_D$. Fix an edge $(r_i^x(u), r_j^y(v))$ added to D_1 in Step 2, for some $i \leq j \in [0, z]$, $x, y \in \{1, 2\}$, and $(u, v) \in E(D^*)$. Since $i \geq \ell(u, v)$, it follows that $S_i(u) \neq S_i(v)$. Additionally, since edge

$(u,v) \in E(D^*) \subseteq E(G_i)$, it follows that $S_i(u) <_i S_i(v)$. Then since $r_i^x(u) \in S_i(u)$ and $r_j^y(v) \in S_j(v) \subseteq S_i(v)$, we conclude that

$$r_i^x(u) <_D r_j^y(v),$$

as desired. □

Construction of DAG D_2.

(1) The vertex set of D_2 will be the same as G, so $V(D_2) = V(G)$.
(2) Fix an index $i \in [0,z]$, and fix a set $S_i^j \in \mathcal{F}_i$, where $j \in [1,k_i]$. Let $H_i^j \subseteq S_i^j \times S_i^j$ be the set of directed, weighted edges specified in Claim 4.5 with respect to set S_i^j. Let $H \subseteq V(G) \times V(G)$ be the directed, weighted set of edges

$$H = \bigcup_{i\in[0,z], j\in[1,k_i]} H_i^j.$$

Let H^R denote the set of weighted edges obtained by reversing the orientation of every edge in H. We add the edges in H^R to D_2.

This completes the construction of graph D_2. Note that for every edge (u,v) in graph D_2, we have that $(v,u) \in E(D_1)$. Then by Claim 4.6, graph D_2 is also a DAG.

Construction of DAG Cover $\mathcal{D}$. We will construct our DAG cover $\mathcal{D}$ by repeating the following (random) procedure $10 \log n$ times:

(1) Construct a collection of sets $\{\mathcal{F}_i\}_{i\in[0,z]}$. (Note that each set $\mathcal{F}_i$ is constructed randomly and inherits the probabilistic guarantees of Lemma 4.1.)
(2) Construct DAGs D_1 and D_2 using the collection of sets $\{\mathcal{F}_i\}_{i\in[0,z]}$, and add D_1 and D_2 to $\mathcal{D}$.

4.2 Size and Time Analysis of DAG Cover $\mathcal{D}$

In this section, we prove that our DAG cover $\mathcal{D}$ has the size claimed in Theorem 1.5, and we prove that our DAG cover can be constructed in the time claimed in Theorem 1.5.

Size Analysis. In Step 2 of the construction of DAG D_1, we add at most $O(z^2)$ edges to D_1 for every edge $(u,v) \in E(D^*) \subseteq E(G)$. Then Step 2 of the construction of DAG D_1 contributes $O(mz^2) = O(m\log^2(nW))$ edges to D_1. By Claim 4.5, the third step of the construction of DAG D_1 contributes at most

$$\begin{aligned}|H| &\le \sum_{i\in[0,z], j\in[1,k_i]} \left|H_i^j\right| \\ &\le O(\log n) \cdot \sum_{i\in[0,z]} \sum_{j\in[1,k_i]} |S_i^j| \\ &= O(\log n) \cdot \sum_{i\in[0,z]} n = O(n\log(nW)\log n)\end{aligned}$$

edges. Then we add at most

$$|E(D_1)| + |E(D_2)| = O(m\log^2 n + n\log^2 n)$$

edges to D_1 and D_2, when $W = O(\text{poly}(n))$. Since we repeatedly construct random DAGs D_1 and D_2 exactly $10\log n$ times, the number of additional edges and DAGs claimed in Theorem 1.5 is established.

Running Time Analysis. To construct our hierarchical decomposition $\{\mathcal{F}\}_i$, we make z calls to Lemma 4.1. This takes $O(\log(nW) \cdot (m\log^2 n + n\log^3 n))$ time. The second step of the construction of DAG D_1 takes $O(mz^2) = O(m\log^2(nW))$ time. To construct all sets of edges H_i^j, where $i \in [0,z]$ and $j \in [1,k_i]$, it takes time $O(\sum_{i\in[0,z], j\in[1,k_i]} |H_i^j|) = O(n\log(nW)\log n)$ by Claim 4.5 and our earlier bound on $\cup_{i,j}|H_i^j|$ in our size analysis. We conclude that the constructions of DAGs D_1 and D_2 take

$$\begin{aligned}&O(\log(nW) \cdot (m\log^2 n + n\log^3 n) + m\log^2(nW)) \\ &\qquad = O(m\log^3 n + n\log^4 n)\end{aligned}$$

time, when $W = O(\text{poly}(n))$. Since we repeatedly construct random DAGs D_1 and D_2 exactly $10\log n$ times, the running time claimed in Theorem 1.5 is established.

4.3 Initial Distortion Analysis

In this section, we make some initial progress towards proving that our DAG cover $\mathcal{D}$ approximately preserves distances in G. First, we verify that for every DAG $D \in \mathcal{D}$, distances in D are at least distances in G.

Claim 4.7 *For each DAG $D \in \mathcal{D}$ and nodes $s,t \in V(G)$,*

$$\text{dist}_G(s,t) \le \text{dist}_D(s,t).$$

Proof. We will prove that for every edge $(u,v) \in E(D)$,

$$\text{dist}_G(u,v) \le w_D(u,v),$$

where w_D is the weight function associated with DAG D. This will immediately imply the stated claim. We split our proof into cases based on which step of the construction we added edge (u,v) to DAG D.

- Edge (u,v) was added in Step 2 of the construction of DAG D_1. In this case, edge (u,v) is of the form

$$(u,v) = (r_i^x(u^*), r_j^y(v^*)),$$

for some $i,j \in [0,z]$ such that $\min(i,j) \ge \ell(u^*,v^*)$, $x,y \in \{1,2\}$ and edge $(u^*,v^*) \in E(D^*)$. Recall that $w_D(u,v) = w_G(u^*,v^*) + d_i + d_j$. Then

$$\begin{aligned}\text{dist}_G(u,v) &= \text{dist}_G(r_i^x(u^*), r_j^y(v^*)) \\ &\le \text{dist}_G(r_i^x(u^*), u^*) + \text{dist}_G(u^*,v^*) + \text{dist}_G(v^*, r_j^y(v^*)) \\ &\le \text{dist}_G(r_i^x(u^*), u^*) + w_G(u^*,v^*) + \text{dist}_G(v^*, r_j^y(v^*)) \\ &\le d_i + w_G(u^*,v^*) + d_j \\ &= w_D(u,v),\end{aligned}$$

as desired.
- Edge (u,v) was added in Step 3 of the construction of DAG D_1 or in Step 2 of the construction of DAG D_2. We will need the following observations:
  - Since edge $(u,v) \in H \cup H^R$, there exists a set $S_i^j \in \mathcal{F}_i$, where $i \in [0,z]$ and $j \in [1,k_i]$, such that $u,v \in S_i^j$.
  - By Property 2 of Claim 4.5, $w(u,v) = d_i$.
  - By Claim 4.2, set S_i^j has weak diameter at most d_i in G.

  Then

$$\text{dist}_G(u,v) \le d_i = w_D(u,v).$$ □

We have shown that distances in our DAG cover are at least distances in G. We now establish the distortion upper bound guarantee of our DAG cover. Let D_1 and D_2 be the random DAGs constructed in subsection 4.1, with associated set family $\{\mathcal{F}_i\}_i$ and graph family $\{G_i\}_i$. The key step in establishing our distortion upper bound will be to prove the following lemma.

Lemma 4.8 *For all $s, t \in V(G)$,*

$$\mathbb{E}[\min(\text{dist}_{D_1}(s,t), \text{dist}_{D_2}(s,t))] = O(\log^4 n) \cdot \text{dist}_G(s,t).$$

Once we prove this lemma, our claimed distortion upper bound in Theorem 1.5 will follow from a simple application of Markov's inequality. The remainder of this section will be devoted to developing the necessary claims and lemmas to prove Lemma 4.8; we will finally prove Lemma 4.8 in section 4.4.

First, we will show that paths in graph G_i have a natural structure with respect to the SCCs $\mathcal{F}_i$ of G_i.

Claim 4.9 *Let π be an $s \rightsquigarrow t$ path in G_i for some $s, t \in V(G)$ and $i \in [0, z]$. There exists a sequence of sets $S_1, \ldots, S_k \subseteq \mathcal{F}_i$ in set family $\mathcal{F}_i$ with the following properties:*

(1) Path π contains exactly one edge $e_j = (u_j, v_j)$ of the form $e_j \in S_j \times S_{j+1}$ for all $j \in [1, k-1]$. Moreover,
- *$\pi[s, u_1] \subseteq S_1$,*
- *$\pi[v_j, u_{j+1}] \subseteq S_{j+1}$ for all $j \in [1, k-2]$, and*
- *$\pi[v_{k-1}, t] \subseteq S_k$.*

(2) $S_{j_1} \neq S_{j_2}$ for all $j_1, j_2 \in [1, k]$ such that $j_1 \neq j_2$.

Proof. Recall that $\mathcal{F}_i$ is the set of all strongly connected components in G_i. Let $u, v \in \pi$ be nodes such that u comes before v in path π. Suppose that $u, v \in \pi \cap S$ for some $S \in \mathcal{F}_i$. This implies that $\pi[u, v] \subseteq S$ because S is an SCC in G_i. Then for every SCC $S \in \mathcal{F}_i$, set S intersects path π in a (possibly empty) contiguous subpath of π. Let sequence $S_1, \ldots, S_k \subseteq \mathcal{F}_i$ be the SCCs in $\mathcal{F}_i$ that have nonempty intersection with π, written so that if $j_1 < j_2 \in [1, k]$ then subpath $\pi \cap S_{j_1}$ comes before subpath $\pi \cap S_{j_2}$ in path π. □

We will now show that if nodes s and t are contained in the same set S in set family $\mathcal{F}_i$, then we can use the edges in H and H^R to upper bound the distances between s and t in D_1 and D_2.

Claim 4.10 *Fix an index $i \in [0, z]$ and nodes $s, t \in S$ for some set $S \in \mathcal{F}_i$ in set family $\mathcal{F}_i$.*

- *If $s <_D t$, then*
$$\text{dist}_{D_1}(s,t) \leq 2d_i.$$
- *if $s >_D t$, then*
$$\text{dist}_{D_2}(s,t) \leq 2d_i.$$

Proof. This claim will follow directly from Claim 4.5 and our construction of DAGs D_1 and D_2. Let $H_S \subseteq S \times S$ be the directed, weighted edges specified in Claim 4.5 with respect to set $S \in \mathcal{F}_i$. By construction, $H_S \subseteq E(D_1)$.

- If $s <_D t$, then
$$\text{dist}_{D_1}(s,t) \leq \text{dist}_{H_S}(s,t) \leq 2d_i,$$
by Property 3 of Claim 4.5.
- Otherwise, $s >_D t$. Let H_S^R be the set of weighted, directed edges obtained by reversing the orientations of the edges in H_S. By construction, $H_S^R \subseteq H^R \subseteq E(D_2)$. Then since $s >_D t$, we can again argue by Property 3 of Claim 4.5 that
$$\text{dist}_{D_2}(s,t) \leq \text{dist}_{H_S^R}(s,t) = \text{dist}_{H_S}(t,s) \leq 2d_i. \qquad \square$$

For an event A, let $\mathbb{1}[A]$ denote the random variable that takes value 1 when event A holds, and value 0 otherwise. The following lemma makes some progress towards proving Lemma 4.8, by achieving an upper bound on the expected distortion between s and t in DAG D_2, multiplied by the binary random variable $\mathbb{1}[s >_D t]$. Notice that if $s <_D t$, then $\text{dist}_{D_2}(s,t) = \infty$, so we cannot expect to get a reasonable upper bound on $\mathbb{E}[\text{dist}_{D_2}(s,t)]$ in general; this is why we upper bound $\mathbb{E}[\text{dist}_{D_2}(s,t) \cdot \mathbb{1}[s >_D t]]$ instead.

Lemma 4.11 *For all $s, t \in V(G)$,*

$$\mathbb{E}[\text{dist}_{D_2}(s,t) \cdot \mathbb{1}[s >_D t]] = O(\log^3 n) \cdot \text{dist}_G(s,t).$$

Proof. Fix a shortest $s \rightsquigarrow t$ path π in G. Let X be the random variable $X = \text{dist}_{D_2}(s,t)$. Let A be the event that $s >_D t$. Let B_i be the event that path π is contained in graph G_i, for all $i \in [0, z]$.

Consider the scenario where event $A \cap B_i$ occurs, i.e., $s >_D t$, and path π is contained in G_i. Then there must exist an SCC $S \in \mathcal{F}_i$ of G_i such that $\pi \subseteq G_i[S]$. If this is not the case, then path π intersects with two or more SCCs in G_i, implying that $s <_D t$, a contradiction. Then we can apply Claim 4.10 to argue that if event $A \cap B_i$ occurs, then $\text{dist}_{D_2}(s,t) \leq 2d_i$. In particular, this implies that

$$\mathbb{E}[X \mid A \cap B_i \cap \overline{B}_{i+1}] \leq 2d_i.$$

Observe that $\Pr[B_0] = 1$ since $\pi \subseteq G = G_0$. Then we can obtain the following inequality

$$\mathbb{E}[X \mid A] \leq \sum_{i=0}^{z} \mathbb{E}[X \mid A \cap B_i \cap \bar{B}_{i+1}] \Pr[B_i \cap \bar{B}_{i+1} \mid A].$$

Additionally, we observe that by Claim 4.2 and the union bound, for all $i \in [0, z]$,

$$\Pr[\bar{B}_{i+1} \mid B_i] \leq \sum_{e \in E(\pi)} O\left(\frac{w(e) \cdot \log^2 n}{d_i}\right) = O\left(\frac{\log^2 n}{d_i}\right) \cdot \text{dist}_G(s,t),$$

since π is an $s \rightsquigarrow t$ shortest path in G. Putting our observations together,

$$\begin{aligned}
&\mathbb{E}[X \cdot \mathbb{1}[s >_D t]] \\
&= \mathbb{E}[X \mid A] \Pr[A] \\
&\leq \left(\sum_{i=0}^{z} \mathbb{E}[X \mid A \cap B_i \cap \bar{B}_{i+1}] \Pr[B_i \cap \bar{B}_{i+1} \mid A]\right) \cdot \Pr[A] \\
&= \sum_{i=0}^{z} \mathbb{E}[X \mid A \cap B_i \cap \bar{B}_{i+1}] \Pr[B_i \cap \bar{B}_{i+1} \cap A] \\
&\leq \sum_{i=0}^{z} \mathbb{E}[X \mid A \cap B_i \cap \bar{B}_{i+1}] \Pr[\bar{B}_{i+1} \mid B_i] \\
&\leq \sum_{i=0}^{z} 2d_i \cdot \Pr[\bar{B}_{i+1} \mid B_i] \\
&\leq \sum_{i=0}^{z} 2d_i \cdot O\left(\frac{\log^2 n}{d_i}\right) \cdot \mathrm{dist}_G(s,t) \\
&= O(z \log^2 n) \cdot \mathrm{dist}_G(s,t) \\
&= O(\log^3 n) \cdot \mathrm{dist}_G(s,t). \qquad \square
\end{aligned}$$

We will need the following technical lemma, which we prove using a similar argument as in Lemma 4.11. Recall that for any index $X \in [0,z]$, $d_X = n^2 W 2^{-X-1}$ is the diameter of the directed low-diameter decomposition that we apply to graph G_X.

Lemma 4.12 *Let $s,t \in V(G)$ be nodes such that s can reach t in G. Let π be an $s \rightsquigarrow t$ shortest path in G. Let $H \subseteq G$ be a subgraph of G such that $\pi \subseteq H$. For each $i \in [0,z]$ and subgraph $J \subseteq G$, let C_i^J be the event that $G_i = J$. Let $X \in [0,z]$ be the (random-valued) index such that $\pi \subseteq G_X$ and $\pi \nsubseteq G_{X+1}$. For each $i \in [0,z]$,*

$$\mathbb{E}\left[d_X | C_i^H\right] \leq c \cdot (z-i+1) \cdot \log^2 n \cdot \mathrm{dist}_G(s,t),$$

for a sufficiently large constant $c > 0$.

Proof. Fix a pair of nodes $s,t \in V(G)$ such that s can reach t in G. Let π be an $s \rightsquigarrow t$ shortest path in G, and let $H \subseteq G$ be a subgraph of G such that $\pi \subseteq H$. We will prove Lemma 4.12 for a fixed index $i \in [0,z]$.

For each $j \in [0,z]$, let B_j be the event that $\pi \subseteq G_j$. We can obtain the following equality:

$$\mathbb{E}[d_X \mid C_i^H] = \sum_{j=i}^{z} d_j \cdot \Pr[B_j \cap \bar{B}_{j+1} \mid C_i^H].$$

Additionally, we observe that by Claim 4.2 and the union bound, for all $j \in [i,z]$,

$$\Pr[\bar{B}_{j+1} \mid B_j \cap C_i^H] \leq \sum_{e \in E(\pi)} O\left(\frac{w(e) \cdot \log^2 n}{d_j}\right) = O\left(\frac{\log^2 n}{d_j}\right) \cdot \mathrm{dist}_G(s,t),$$

since π is an $s \rightsquigarrow t$ shortest path in G. Putting our observations together,

$$\begin{aligned}
\mathbb{E}[d_X \mid C_i^H] &= \sum_{j=i}^{z} d_j \cdot \Pr[B_j \cap \bar{B}_{j+1} \mid C_i^H] \\
&= \sum_{j=i}^{z} d_j \cdot \Pr[\bar{B}_{j+1} \mid B_j \cap C_i^H] \Pr[B_j \mid C_i^H] \\
&\leq \sum_{j=i}^{z} d_j \cdot \Pr[\bar{B}_{j+1} \mid B_j \cap C_i^H] \\
&\leq \sum_{j=i}^{z} d_j \cdot O\left(\frac{\log^2 n}{d_j}\right) \cdot \mathrm{dist}_G(s,t) \\
&\leq c \cdot (z-i+1) \cdot \log^2 n \cdot \mathrm{dist}_G(s,t),
\end{aligned}$$

for a sufficiently large constant $c > 0$. $\square$

In Lemma 4.11, we gave an upper bound on the expectation of random variable $\mathrm{dist}_{D_2}(s,t) \cdot \mathbb{1}[s >_D t]$, for a pair of nodes $s,t \in V(G)$. This proof made heavy use of the fact that this random variable took value 0 when $s <_D t$. Our next goal will be to give an upper bound on the expectation of the random variable $\mathrm{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t]$. We will need to introduce some additional notation first.

Additional Notation. For all nodes $s,t \in V(G)$ such that s can reach t in G (i.e., $(s,t) \in TC(G)$), we fix an $s \rightsquigarrow t$ shortest path $\pi_{s,t}$ in G. We may assume without loss of generality that our collection of paths $\{\pi_{s,t}\}_{(s,t) \in TC(G)}$ is *consistent*. Formally, this means that for any two paths $\pi_{s,t}$ and $\pi_{s',t'}$ in our collection, and for any two vertices $u,v \in V(G)$, if $u,v \in \pi_{s,t} \cap \pi_{s',t'}$ and node u precedes node v in both $\pi_{s,t}$ and $\pi_{s',t'}$, then we have that $\pi_{s,t}[u,v] = \pi_{s',t'}[u,v]$.

For all nodes $s,t \in V(G)$ such that s can reach t in G, and for each $i \in [0,z]$, let $B_i^{s,t}$ be the event that path $\pi_{s,t}$ is contained in G_i. For every subgraph $H \subseteq G$, let C_i^H be the event that $G_i = H$.

The following lemma formally states what we will prove about the expected size of the random variable $\mathrm{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t]$.

Lemma 4.13 *Let $s,t \in V(G)$, and let $i \in [0,z]$. Let $H \subseteq G$ be a subgraph of G such that $\pi_{s,t} \subseteq H$. Then*

$$\mathbb{E}\left[\mathrm{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t] \mid C_i^H\right] \leq c \cdot (z-i+1)^2 \cdot \log^2 n \cdot \mathrm{dist}_G(s,t),$$

for a sufficiently large constant $c > 1$.

We prove Lemma 4.13 by an induction argument in the arXiv version of our paper [6]. We finish the proof of Theorem 1.5 in Section 4.4.

4.4 Finishing the Proof of Theorem 1.5

With Lemma 4.13 in hand, we can now easily finish our proof of Theorem 1.5.

Claim 4.14 *For all $s,t \in V(G)$,*

$$\mathbb{E}[\mathrm{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t]] = O(\log^4 n) \cdot \mathrm{dist}_G(s,t).$$

Proof. Recall that C_0^G is the event that $G_0 = G$. Notice that by our construction, $G_0 = G$ and shortest path $\pi_{s,t}$ satisfies $\pi_{s,t} \subseteq G_0$, unconditionally. Then by Lemma 4.13,

$$\begin{aligned}\mathbb{E}[\text{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t]] &= \mathbb{E}[\text{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t] \mid C_0^G] \\ &\leq O(z^2 \log^2 n) \cdot \text{dist}_G(s,t).\end{aligned}$$

Since $z = O(\log n)$, the claim follows. □

We are ready to prove Lemma 4.8, which we restate below.

Lemma 4.8 *For all $s, t \in V(G)$,*

$$\mathbb{E}[\min(\text{dist}_{D_1}(s,t), \text{dist}_{D_2}(s,t))] = O(\log^4 n) \cdot \text{dist}_G(s,t).$$

Proof.

$$\begin{aligned}&\mathbb{E}[\min(\text{dist}_{D_1}(s,t), \text{dist}_{D_2}(s,t))] \\ &\leq \mathbb{E}[\text{dist}_{D_1}(s,t) \cdot \mathbb{1}[s <_D t]] + \mathbb{E}[\text{dist}_{D_2}(s,t) \cdot \mathbb{1}[s >_D t]] \\ &\leq O(\log^4 n) \cdot \text{dist}_G(s,t),\end{aligned}$$

where the final inequality follows from Lemma 4.11 and Claim 4.14. □

We can now finish the distortion analysis and the proof of Theorem 1.5.

Proof of Theorem 1.5. We have finished the running time analysis and size analysis for Theorem 1.5. What remains is to prove that with high probability, for all $s, t \in V(G)$, there exists $D \in \mathcal{D}$ such that

$$\text{dist}_D(s,t) \leq O(\log^4 n) \cdot \text{dist}_G(s,t).$$

Fix a pair of nodes $s, t \in V(G)$. Since $\mathbb{E}[\min(\text{dist}_{D_1}(s,t), \text{dist}_{D_2}(s,t))] = O(\log^4 n) \cdot \text{dist}_G(s,t)$ by Lemma 4.8, by Markov's inequality we conclude that with probability at least $1/2$,

$$\min(\text{dist}_{D_1}(s,t), \text{dist}_{D_2}(s,t)) = O(\log^4 n) \cdot \text{dist}_G(s,t).$$

Then with probability at least

$$1 - \left(\frac{1}{2}\right)^{10 \log n} \geq 1 - n^{-10},$$

there exists a DAG $D \in \mathcal{D}$ such that $\text{dist}_D(s,t) \leq \text{dist}_G(s,t)$. Then by applying the union bound over all pairs of nodes $s, t \in V(G)$, we conclude that $\mathcal{D}$ is an $O(\log^4 n)$-distance-preserving DAG cover of G with high probability. □

5 Open Problems

The main problem left open by our work is closing the gap between our upper and lower bounds for DAG covers with $\tilde{O}(m)$ additional edges. Our upper bound is polylogarithmic in both the distortion and the number of DAGs, while our lower bound is for exact distances and a polynomial number of DAGs. We conjecture that the lower bound can be improved to handle approximate distances, as well as a larger polynomial number of DAGs. Another open problem is to find concrete applications of our DAG cover algorithm. We suspect that such applications exist, given the wide-reaching applications of the analogous undirected constructions. Another open problem is to extend our DAG cover algorithm to various settings such as the distributed, parallel, online, dynamic, and streaming settings, since such settings have been fruitful for the analogous undirected constructions. Lastly, it would be interesting to consider the "Steiner" version of this problem where vertices, in addition to edges, are allowed to be added.

Acknowledgements

We would like to thank Aaron Bernstein, Shimon Kogan, and Merav Parter for the conversation that initiated this work, as well as additional fruitful conversations about how to define the problem.

References

[1] Reyan Ahmed, Greg Bodwin, Faryad Darabi Sahneh, Keaton Hamm, Mohammad Javad Latifi Jebelli, Stephen Kobourov, and Richard Spence. 2020. Graph spanners: a tutorial review. *Comput. Sci. Rev.* 37 (2020), 100253, 30. doi:10.1016/j.cosrev.2020.100253

[2] Shyan Akmal and Nicole Wein. 2023. A local-to-global theorem for congested shortest paths. In *31st annual European Symposium on Algorithms*. LIPIcs. Leibniz Int. Proc. Inform., Vol. 274. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 8, 17. doi:10.4230/lipics.esa.2023.8

[3] Noga Alon, Richard M. Karp, David Peleg, and Douglas West. 1995. A graph-theoretic game and its application to the k-server problem. *SIAM J. Comput.* 24, 1 (1995), 78–100. doi:10.1137/S0097539792224474

[4] Saeed Akhoondian Amiri and Julian Wargalla. 2020. Disjoint shortest paths with congestion on dags. *arXiv preprint arXiv:2008.08368* (2020).

[5] Sunil Arya, Gautam Das, David M Mount, Jeffrey S Salowe, and Michiel Smid. 1995. Euclidean spanners: short, thin, and lanky. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*. 489–498.

[6] Sepehr Assadi, Gary Hoppenworth, and Nicole Wein. 2025. Covering Approximate Shortest Paths with DAGs. *arXiv preprint arXiv:2504.11256* (2025).

[7] Baruch Awerbuch, Shay Kutten, and David Peleg. 1991. Efficient deadlock-free routing. In *Proceedings of the tenth annual ACM symposium on Principles of distributed computing*. 177–188.

[8] Baruch Awerbuch, Shay Kutten, and David Peleg. 1994. On buffer-economical store-and-forward deadlock prevention. *IEEE transactions on communications* 42, 11 (1994), 2934–2937.

[9] Baruch Awerbuch and David Peleg. 1992. Routing with polynomial communication-space trade-off. *SIAM J. Discrete Math.* 5, 2 (1992), 151–162. doi:10.1137/0405013

[10] Baruch Awerbuch and David Peleg. 1995. Online tracking of mobile users. *Journal of the ACM (JACM)* 42, 5 (1995), 1021–1058.

[11] Lorenzo Balzotti. 2022. Non-crossing shortest paths are covered with exactly four forests. *arXiv preprint arXiv:2210.13036* (2022).

[12] Yair Bartal. 1996. Probabilistic approximation of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science (Burlington, VT, 1996)*. IEEE Comput. Soc. Press, Los Alamitos, CA, 184–193. doi:10.1109/SFCS.1996.548477

[13] Yair Bartal. 1999. On approximating arbitrary metrices by tree metrics. In *STOC '98 (Dallas, TX)*. ACM, New York, 161–168.

[14] Yair Bartal, Nova Fandina, and Ofer Neiman. 2019. Covering metric spaces by few trees. In *46th International Colloquium on Automata, Languages, and Programming*. LIPIcs. Leibniz Int. Proc. Inform., Vol. 132. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 20, 16.

[15] Yair Bartal, Nova Fandina, and Seeun William Umboh. 2020. Online probabilistic metric embedding: a general framework for bypassing inherent bounds. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Philadelphia, PA, 1538–1557.

[16] Yair Bartal, Ora Nova Fandina, and Ofer Neiman. 2022. Covering metric spaces by few trees. *J. Comput. System Sci.* 130 (2022), 26–42. doi:10.1016/j.jcss.2022.06.001

[17] Yair Bartal, Nathan Linial, Manor Mendel, and Assaf Naor. 2005. On metric Ramsey-type phenomena. *Ann. of Math. (2)* 162, 2 (2005), 643–709. doi:10.4007/annals.2005.162.643

[18] Aaron Bernstein, Greg Bodwin, and Nicole Wein. 2024. Are there graphs whose shortest path structure requires large edge weights? In *15th Innovations in Theoretical Computer Science Conference*. LIPIcs. Leibniz Int. Proc. Inform., Vol. 287. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 12, 22. doi:10.4230/lipics.itcs.2024.12

[19] Aaron Bernstein, Danupon Nanongkai, and Christian Wulff-Nilsen. 2022. Negative-weight single-source shortest paths in near-linear time. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science—FOCS 2022*. IEEE Computer Soc., Los Alamitos, CA, 600–611.

[20] Aaron Bernstein and Nicole Wein. 2023. Closing the gap between directed hopsets and shortcut sets. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on*

Discrete Algorithms (SODA). SIAM, Philadelphia, PA, 163–182. doi:10.1137/1.9781611977554.ch7

[21] Dietmar Berwanger, Anuj Dawar, Paul Hunter, and Stephan Kreutzer. 2006. DAG-width and parity games. In *STACS 2006*. Lecture Notes in Comput. Sci., Vol. 3884. Springer, Berlin, 524–536. doi:10.1007/11672142_43

[22] Guy E. Blelloch, Yan Gu, and Yihan Sun. 2017. Efficient construction of probabilistic tree embeddings. In *44th International Colloquium on Automata, Languages, and Programming*. LIPIcs. Leibniz Int. Proc. Inform., Vol. 80. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 26, 14.

[23] Guy E Blelloch, Anupam Gupta, and Kanat Tangwongsan. 2012. Parallel probabilistic tree embeddings, k-median, and buy-at-bulk network design. In *Proceedings of the twenty-fourth annual ACM symposium on Parallelism in algorithms and architectures*. 205–213.

[24] Greg Bodwin. 2017. Linear size distance preservers. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 600–615.

[25] Greg Bodwin. 2019. On the structure of unique shortest paths in graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Philadelphia, PA, 2071–2089. doi:10.1137/1.9781611975482.125

[26] Greg Bodwin and Gary Hoppenworth. 2023. Folklore sampling is optimal for exact hopsets: confirming the $\sqrt{n}$ barrier. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science—FOCS 2023*. IEEE Computer Soc., Los Alamitos, CA, 701–720. doi:10.1109/FOCS57990.2023.00046

[27] Karl Bringmann, Alejandro Cassis, and Nick Fischer. 2023. Negative-weight single-source shortest paths in near-linear time: now faster! In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science—FOCS 2023*. IEEE Computer Soc., Los Alamitos, CA, 515–538. doi:10.1109/FOCS57990.2023.00038

[28] Karl Bringmann, Alejandro Cassis, and Nick Fischer. 2023. Negative-Weight Single-Source Shortest Paths in Near-Linear Time: Now Faster!. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 515–538.

[29] Karl Bringmann, Nick Fischer, Bernhard Haeupler, and Rustam Latypov. 2025. Near-Optimal Directed Low-Diameter Decompositions. *arXiv preprint arXiv:2502.05687* (2025).

[30] Massimo Cairo, Roberto Grossi, and Romeo Rizzi. 2016. New bounds for approximating extremal distances in undirected graphs. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, 363–376. doi:10.1137/1.9781611974331.ch27

[31] T.-H. Hubert Chan, Anupam Gupta, Bruce M. Maggs, and Shuheng Zhou. 2016. On hierarchical routing in doubling metrics. *ACM Trans. Algorithms* 12, 4 (2016), Art. 55, 22. doi:10.1145/2915183

[32] Hsien-Chih Chang, Jonathan Conroy, Hung Le, Lazar Milenkovic, Shay Solomon, and Cuong Than. 2023. Covering planar metrics (and beyond): O (1) trees suffice. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2231–2261.

[33] Moses Charikar, Chandra Chekuri, Ashish Goel, and Sudipto Guha. 1999. Rounding via trees: deterministic approximation algorithms for group Steiner trees and k-median. In *STOC '98 (Dallas, TX)*. ACM, New York, 114–123.

[34] Daniel Cizma and Nati Linial. 2022. Geodesic geometry on graphs. *Discrete Comput. Geom.* 68, 1 (2022), 298–347. doi:10.1007/s00454-021-00345-w

[35] Daniel Cizma and Nati Linial. 2023. Irreducible nonmetrizable path systems in graphs. *J. Graph Theory* 102, 1 (2023), 5–14.

[36] Michael B. Cohen, Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup B. Rao, and Shen Chen Xu. 2014. Solving SDD linear systems in nearly $m \log^{1/2} n$ time. In *STOC'14—Proceedings of the 2014 ACM Symposium on Theory of Computing*. ACM, New York, 343–352.

[37] Sabine Cornelsen, Maximilian Pfister, Henry Förster, Martin Gronemann, Michael Hoffmann, Stephen Kobourov, and Thomas Schneck. 2022. Drawing shortest paths in geodetic graphs. *J. Graph Algorithms Appl.* 26, 3 (2022), 353–361. doi:10.7155/jgaa.00598

[38] Michael Elkin and Ofer Neiman. 2019. Linear-Size Hopsets with Small Hopbound, and Constant-Hopbound Hopsets in RNC. In *The 31st ACM on Symposium on Parallelism in Algorithms and Architectures, SPAA 2019, Phoenix, AZ, USA, June 22-24, 2019*, Christian Scheideler and Petra Berenbrink (Eds.). ACM, 333–341. doi:10.1145/3323165.3323177

[39] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. 2004. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. System Sci.* 69, 3 (2004), 485–497. doi:10.1016/j.jcss.2004.04.011

[40] Arnold Filtser and Hung Le. 2022. Locality-sensitive orderings and applications to reliable spanners. In *STOC '22—Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, New York, 1066–1079.

[41] Jeremy T. Fineman. 2024. Single-source shortest paths with negative real weights in $(mn^{8/9})$ time. In *STOC'24—Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. ACM, New York, 3–14. doi:10.1145/3618260.3649614

[42] Sebastian Forster, Gramoz Goranci, and Monika Henzinger. 2021. Dynamic maintenance of low-stretch probabilistic tree embeddings with applications. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. [Society for Industrial and Applied Mathematics (SIAM)], Philadelphia, PA, 1226–1245. doi:10.1137/1.9781611976465.75

[43] Stephan Friedrichs and Christoph Lenzen. 2018. Parallel metric tree embedding based on an algebraic view on Moore-Bellman-Ford. *J. ACM* 65, 6 (2018), Art. 43, 55. doi:10.1145/3231591

[44] Naveen Garg, Goran Konjevod, and R. Ravi. 2000. A polylogarithmic approximation algorithm for the group Steiner tree problem. Vol. 37. 66–84. doi:10.1006/jagm.2000.1096 Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (San Francisco, CA, 1998).

[45] Mohsen Ghaffari and Christoph Lenzen. 2014. Near-optimal distributed tree embedding. In *Distributed computing*. Lecture Notes in Comput. Sci., Vol. 8784. Springer, Heidelberg, 197–211. doi:10.1007/978-3-662-45174-8_14

[46] Anupam Gupta, Amit Kumar, and Rajeev Rastogi. 2005. Traveling with a pez dispenser (or, routing issues in mpls). *SIAM J. Comput.* 34, 2 (2005), 453–474.

[47] Bernhard Haeupler, D. Ellis Hershkowitz, and Goran Zuzic. 2021. Tree embeddings for hop-constrained network design. In *STOC '21—Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. ACM, New York, 356–369. doi:10.1145/3406325.3451053

[48] Chris Harrelson, Kirsten Hildrum, and Satish Rao. 2003. A polynomial-time tree decomposition to minimize congestion. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*. 34–43.

[49] William Hesse. 2003. Directed graphs requiring large numbers of shortcuts.. In *SODA*. Citeseer, 665–669.

[50] Shang-En Huang and Seth Pettie. 2019. Thorup-Zwick emulators are universally optimal hopsets. *Inf. Process. Lett.* 142 (2019), 9–13. doi:10.1016/j.ipl.2018.10.001

[51] Yufan Huang, Peter Jin, and Kent Quanrud. 2024. Faster single-source shortest paths with negative real weights via proper hop distance. *arXiv preprint arXiv:2407.04872* (2024).

[52] Omri Kahalon, Hung Le, Lazar Milenković, and Shay Solomon. 2022. Can't see the forest for the trees: Navigating metric spaces by bounded hop-diameter spanners. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*. 151–162.

[53] Richard M Karp. 1989. A 2k-competitive algorithm for the circle. *Manuscript, August* 5 (1989), 11.

[54] Marek Karpinski, Andrzej Lingas, and Dzmitry Sledneu. 2013. Optimal cuts and partitions in tree metrics in polynomial time. *Inform. Process. Lett.* 113, 12 (2013), 447–451. doi:10.1016/j.ipl.2013.03.009

[55] Ken-ichi Kawarabayashi and Anastasios Sidiropoulos. 2022. Embeddings of planar quasimetrics into directed ℓ_1 and polylogarithmic approximation for Directed Sparsest-Cut. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science—FOCS 2021*. IEEE Computer Soc., Los Alamitos, CA, 480–491. doi:10.1109/FOCS52979.2021.00055

[56] Maleq Khan, Fabian Kuhn, Dahlia Malkhi, Gopal Pandurangan, and Kunal Talwar. 2008. Efficient distributed approximation algorithms via probabilistic tree embeddings. In *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*. 263–272.

[57] Jon Kleinberg and Éva Tardos. 2002. Approximation algorithms for classification problems with pairwise relationships: metric labeling and Markov random fields. *J. ACM* 49, 5 (2002), 616–639. doi:10.1145/585265.585268

[58] Shimon Kogan and Merav Parter. 2022. New diameter-reducing shortcuts and directed hopsets: breaking the $O(\sqrt{n})$ barrier. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. [Society for Industrial and Applied Mathematics (SIAM)], Philadelphia, PA, 1326–1340. doi:10.1137/1.9781611977073.55

[59] I. Krasikov and S. D. Noble. 2004. Finding next-to-shortest paths in a graph. *Inform. Process. Lett.* 92, 3 (2004), 117–119. doi:10.1016/j.ipl.2004.06.020

[60] Willian Lochet. 2021. A polynomial time algorithm for the k-disjoint shortest paths problem. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. [Society for Industrial and Applied Mathematics (SIAM)], Philadelphia, PA, 169–178. doi:10.1137/1.9781611976465.12

[61] Manor Mendel and Assaf Naor. 2007. Ramsey partitions and proximity data structures. *J. Eur. Math. Soc. (JEMS)* 9, 2 (2007), 253–275. doi:10.4171/JEMS/79

[62] Manor Mendel and Chaya Schwob. 2009. Fast CKR Partitions of Sparse Graphs. *Chicago Journal OF Theoretical Computer Science* 2 (2009), 1–18.

[63] Harald Racke. 2002. Minimizing congestion in general networks. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.* IEEE, 43–52.

[64] Harald Räcke. 2008. Optimal hierarchical decompositions for congestion minimization in networks. In *STOC'08*. ACM, New York, 255–263. doi:10.1145/1374376.1374415

[65] Mikkel Thorup and Uri Zwick. 2005. Approximate distance oracles. *Journal of the ACM (JACM)* 52, 1 (2005), 1–24.

[66] Virginia Vassilevska Williams, Yinzhan Xu, and Zixuan Xu. 2024. Simpler and higher lower bounds for shortcut sets. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, Philadelphia, PA, 2643–2656. doi:10.1137/1.9781611977912.94

Received 2024-11-04; accepted 2025-02-01