Approximation Algorithms for Directed Weighted Spanners

Elena Grigorescu[*] Nithish Kumar[†] Young-San Lin[‡]

July 10, 2023

Abstract

In the *pairwise weighted spanner* problem, the input consists of a weighted directed graph on n vertices, where each edge is assigned both a *cost* and a *length*. Furthermore, we are given k terminal vertex pairs and a distance constraint for each pair. The goal is to find a minimum-cost subgraph in which the distance constraints are satisfied. A more restricted variant of this problem was shown to be $O(2^{\log^{1-\varepsilon} n})$-hard to approximate under a standard complexity assumption, by Elkin and Peleg (Theory of Computing Systems, 2007). This general formulation captures many well-studied network connectivity problems, including spanners, distance preservers, and Steiner forests.

We study the weighted spanner problem, in which the edges have positive *integral* lengths of magnitudes that are *polynomial* in n, while the costs are *arbitrary* non-negative rational numbers. Our results include the following in the classical offline setting:

- An $\tilde{O}(n^{4/5+\varepsilon})$-approximation algorithm for the pairwise weighted spanner problem. When the edges have unit costs and lengths, the best previous algorithm gives an $\tilde{O}(n^{3/5+\varepsilon})$-approximation, due to Chlamtáč, Dinitz, Kortsarz, and Laekhanukit (Transactions on Algorithms, 2020).

- An $\tilde{O}(n^{1/2+\varepsilon})$-approximation algorithm for the weighted spanner problem when the terminal pairs consist of *all* vertex pairs and the distances must be preserved *exactly*. When the edges have unit costs and arbitrary positive lengths, the best previous algorithm gives an $\tilde{O}(n^{1/2})$-approximation for the all-pair spanner problem, due to Berman, Bhattacharyya, Makarychev, Raskhodnikova, and Yaroslavtsev (Information and Computation, 2013).

We also prove the first results for the weighted spanners in the *online* setting. In the online setting, the terminal vertex pairs arrive one at a time, in an online fashion, and edges are required to be added irrevocably to the solution in order to satisfy the distance constraints, while approximately minimizing the cost. Our results include the following:

- An $\tilde{O}(k^{1/2+\varepsilon})$-competitive algorithm for the online pairwise weighted spanner problem. The state-of-the-art results are an $\tilde{O}(n^{4/5})$-competitive algorithm when edges have unit costs and arbitrary positive lengths, and a $\min\{\tilde{O}(k^{1/2+\varepsilon}), \tilde{O}(n^{2/3+\varepsilon})\}$-competitive algorithm when edges have unit costs and lengths, due to Grigorescu, Lin, and Quanrud (APPROX, 2021).

- An $\tilde{O}(k^{\varepsilon})$-competitive algorithm for the online single-source (or single-sink) weighted spanner problem. Without distance constraints, this problem is equivalent to the online directed Steiner tree problem. The best previous algorithm for online directed Steiner trees is an $\tilde{O}(k^{\varepsilon})$-competitive algorithm, due to Chakrabarty, Ene, Krishnaswamy, and Panigrahi (SICOMP, 2018).

Our online results also imply efficient approximation algorithms for the corresponding offline problems. To the best of our knowledge, these are the first approximation (online) polynomial-time algorithms with sublinear approximation (competitive) ratios for the weighted spanner problems.

---

[*]Purdue University. E-mail: elena-g@purdue.edu. Supported in part by NSF CCF-1910659, NSF CCF-1910411, and NSF CCF-2228814.

[†]Purdue University. E-mail: kumar410@purdue.edu. Supported in part by NSF CCF-1910411, and NSF CCF-2228814.

[‡]Melbourne Business School. E-mail: y.lin@mbs.edu.

1 Introduction

We study a multi-commodity problem in directed graphs, which we call the *pairwise weighted spanner* problem. In this problem, we are given a directed simple graph $G = (V, E)$ with n vertices, and a set of k terminal pairs $D \subseteq V \times V$. Furthermore, each edge $e \in E$ is associated with a *cost* given by the function $c : E \to \mathbb{R}_{\geq 0}$ and a *length* given by the function $\ell : E \to \mathbb{R}_{\geq 0}$. We say that the graph has unit lengths if $\ell(e) = 1$ (respectively, the graph has unit costs if $c(e) = 1$) for all $e \in E$. Each pair $(s, t) \in D$ is also associated with a target distance given by a function $Dist : D \to \mathbb{R}_{\geq 0}$. Let $H = (V(H), E(H))$ be a subgraph of G and $d_H(s, t)$ denote the *distance* from s to t in H, i.e., the total length of a shortest $s \rightsquigarrow t$ path of edges in $E(H)$. The cost of H is $\sum_{e \in E(H)} c(e)$. The goal is to find a *minimum-cost subgraph H* of G such that the distance from s to t is at most $Dist(s, t)$, namely, $d_H(s, t) \leq Dist(s, t)$ for each $(s, t) \in D$.

The pairwise weighted spanner problem captures many network connectivity problems and is motivated by common scenarios, such as constructing an electricity or an internet network, which requires not only cost minimization but also a delivery time tolerance for the demands. Each edge is thus associated with two "weights" in this setting: the cost and the delivery time. This general formulation has been studied under many variants: when the edges have general lengths and unit costs, one may ask for sparse subgraphs that *exactly* maintain pairwise distances, i.e., *distance preservers*, or for sparse subgraphs that *approximately* maintain pairwise distances, i.e., *spanners*; when the edges have general costs and unit lengths, one may ask for cheap subgraphs that maintain pairwise connectivity, i.e., *Steiner forests*. Spanners and distance preservers are well-studied objects, which have found applicability in domains such as distributed computation [7, 49], data structures [4, 55], routing schemes [20, 47, 50, 52], approximate shortest paths [8, 24, 25], distance oracles [8, 17, 48], and property testing [6, 11]. Similarly, Steiner forests have been studied in the context of multicommodity network design [30, 33], mechanism design and games [16, 42, 43, 53], computational biology [40, 51], and computational geometry [9, 13].

A slightly more special case of the pairwise weighted spanner problem was originally proposed by Kortsarz [44] and Elkin and Peleg [27], where it was called the *weighted s-spanner* problem. The precise goal in [27, 44] is to find a minimum-cost subgraph that connects *all* the pairs of vertices in G, and each $Dist(s, t) = s \cdot d_G(s, t)$ for some integer s called the *stretch* of the spanner. The work of [27] presents a comprehensive list of inapproximability results for different variants of sparse s-spanners. Even in the special case where edges have unit costs (i.e., the directed s-spanner problem defined below), the problem is hard to approximate within a factor of $O(2^{\log^{1-\varepsilon} n})$ unless $NP \subseteq DTIME(n^{\mathrm{polylog}\, n})$.

In the case when the edges have unit costs, the weighted s-spanner problem is called the *directed s-spanner* problem. For low-stretch spanners, when $s = 2$, there is a tight $\Theta(\log n)$-approximate algorithm [26, 44]; with unit lengths and costs, when $s = 3, 4$, there are $\tilde{O}(n^{1/3})$-approximation algorithms [10, 23]. For $s > 4$ with general lengths, the best known approximation is $\tilde{O}(n^{1/2})$ [10].

The *pairwise spanner* problem considers graphs with unit edge costs, D can be any subset of $V \times V$, and the target distances are general. The state-of-the-art is $\tilde{O}(n^{4/5})$-approximate for general lengths [32] and $\min\{\tilde{O}(k^{1/2+\varepsilon}), \tilde{O}(n^{3/5+\varepsilon})\}$-approximation for unit lengths [19, 32].

When the target distances are infinite and the edges have unit lengths, the pairwise weighted spanner problem captures the *directed Steiner forest* problem. For the directed Steiner forest problem, there is an $\min\{\tilde{O}(k^{1/2+\varepsilon}), \tilde{O}(n^{2/3+\varepsilon})\}$-approximate algorithm for general costs [10, 18] and an $\tilde{O}(n^{4/7})$-approximate algorithm for unit costs [1].

The extreme case when there is only one terminal pair, namely the *restricted shortest path* problem, is NP-hard, and it admits a fully polynomial-time approximation scheme (FPTAS)[1] [36, 45].

---

[1] An FPTAS for a minimization problem minimizes the objective within a factor of $1 + \varepsilon$ in polynomial time. The polynomial is in terms of the input size of the problem and $1/\varepsilon$.

1.1 Our contributions

1.1.1 Pairwise weighted spanners

To the best of our knowledge, none of the variants studied in the literature gives efficient sublinear-factor approximation algorithms for the pairwise weighted spanner problem, even in the case of unit edge length. Our main result for pairwise weighted spanners is stated as follows and proved in Section 2.

Definition 1.1. PAIRWISE WEIGHTED SPANNER
    Instance: A directed graph $G = (V, E)$ with n vertices and edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \mathsf{poly}(n)\}$, and a set $D \subseteq V \times V$ of vertex pairs and their corresponding pairwise distance bounds $Dist : D \to \mathbb{Q}_{\geq 0}$ (where $Dist(s, t) \geq d_G(s, t)$) for every terminal pair $(s, t) \in D$.
    Objective: Find a min-cost subgraph H of G such that $d_H(s, t) \leq Dist(s, t)$ for all $(s, t) \in D$.

Theorem 1.2. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized algorithm for* PAIRWISE WEIGHTED SPANNER *with approximation ratio $\tilde{O}(n^{4/5+\varepsilon})$, which succeeds in resolving all pairs in D with high probability.*[2]

The PAIRWISE WEIGHTED SPANNER problem is equivalent to the problem of finding a minimum-cost Steiner forest under pairwise distance constraints, and hence our result is the first polynomial-time $o(n)$-approximate algorithm for the directed Steiner forests with distance constraints. This problem is hard to approximate within a factor of $O(2^{\log^{1-\varepsilon} n})$ unless $NP \subseteq DTIME(n^{\mathsf{polylog}\, n})$ even for the special case when all vertex pairs are required to be connected and the stretch $s \geq 5$ [44].

1.1.2 All-pair weighted distance preservers

When the target distances are the distances in the given graph G, the spanner problem captures the *distance preserver* problem. When edges have unit costs, there exists a distance preserver of cost $O(n)$ if the number of the source vertices is $O(n^{1/3})$ [12]. When edges have unit costs and lengths, the state-of-the-art result is $\tilde{O}(n^{3/5+\varepsilon})$-approximate [19]. We consider the case where the terminal set consists of *all* vertex pairs and the subgraph is required to preserve the distances of all vertex pairs. This problem is called ALL-PAIR WEIGHTED DISTANCE PRESERVER. Our result for this problem is stated as follows and proved in Section 3.

Definition 1.3. ALL-PAIR WEIGHTED DISTANCE PRESERVER
    Instance: A directed graph $G = (V, E)$ with edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \mathsf{poly}(n)\}$.
    Objective: Find a min-cost subgraph H of G such that $d_H(s, t) = d_G(s, t)$, for all $(s, t) \in V \times V$.

Theorem 1.4. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized algorithm for* ALL-PAIR WEIGHTED DISTANCE PRESERVER *with approximation ratio $\tilde{O}(n^{1/2+\varepsilon})$, which succeeds in resolving all pairs in $V \times V$ with high probability.*

Beside distance preservers, there are other previous special-case results for the all-pair weighted spanner problem. When edges have unit costs, the state-of-the-art is an $\tilde{O}(n^{1/2})$-approximation algorithm [10]. When there are no distance constraints, this problem is termed the *minimum strongly connected subgraph* problem and is equivalent to the all-pair Steiner forest problem. This problem is NP-hard and does not admit a polynomial-time approximation scheme if $NP \neq P$ [39]. The best algorithm is a $3/2$-approximation [54].

1.1.3 Online weighted spanners

Next, we turn to online weighted spanners. In the online problem, the directed graph, the edge lengths, and the edge costs are given offline. The vertex pairs and the corresponding target distances arrive one at a time, in an online fashion, at each time stamp. The algorithm must irrevocably select edges at each time stamp

---

[2]Throughout our discussion, when we say high probability we mean probability at least $1 - 1/n$.

and the goal is to minimize the cost, subject to the target distance constraints. We call this problem the ONLINE PAIRWISE WEIGHTED SPANNER problem. For notation convenience, the vertex pair (s_i, t_i) denotes the i-th pair that arrives online.

Definition 1.5. ONLINE PAIRWISE WEIGHTED SPANNER

Instance: A directed graph $G = (V, E)$ with edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \text{poly}(n)\}$, and vertex pairs $D = \{(s_i, t_i) \in V \times V \mid i \in [k]\}$ (k is unknown) with their corresponding pairwise distance bounds $Dist(s_i, t_i) \in \mathbb{Q}_{\geq 0}$ (where $Dist(s_i, t_i) \geq d_G(s_i, t_i)$) arrive online one at a time.

Objective: Upon the arrival of (s_i, t_i) with $Dist(s_i, t_i)$, construct a min-cost subgraph H of G such that $d_H(s_i, t_i) \leq Dist(s_i, t_i)$ by irrevocably adding edges from E.

The performance of an online algorithm is measured by its *competitive ratio*, namely the ratio between the cost of the online solution and that of an optimal offline solution. With unit edge costs, the best algorithm is $\tilde{O}(n^{4/5})$-competitive; with unit edge costs and lengths, the state-of-the-art is $\min\{\tilde{O}(k^{1/2+\varepsilon}), \tilde{O}(n^{2/3+\varepsilon})\}$-competitive [32]. Our result for ONLINE PAIRWISE WEIGHTED SPANNER is stated as follows and proved in Section 4.

Theorem 1.6. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized online algorithm for* ONLINE PAIRWISE WEIGHTED SPANNER *with competitive ratio $\tilde{O}(k^{1/2+\varepsilon})$, which succeeds in resolving all pairs in D with high probability.*

In a special case of PAIRWISE WEIGHTED SPANNER where the source vertex $s \in V$ is fixed, we call this problem SINGLE-SOURCE WEIGHTED SPANNER. Without distance constraints, this problem is equivalent to the directed Steiner tree problem.[3] The best algorithm for the directed Steiner tree problem is $O(k^\varepsilon)$-approximate [15].

Definition 1.7. SINGLE-SOURCE WEIGHTED SPANNER

Instance: A directed graph $G = (V, E)$ with edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \text{poly}(n)\}$, and a set $D \subseteq \{s\} \times V$ of vertex pairs and their corresponding pairwise distance bounds $Dist : D \to \mathbb{Q}_{\geq 0}$ (where $Dist(s, t) \geq d_G(s, t)$ for every terminal pair $(s, t) \in D$).

Objective: Find a min-cost subgraph H of G such that $d_H(s, t) \leq Dist(s, t)$ for all $(s, t) \in D$.

When $D \subseteq \{s\} \times V$, a single-source weighted spanner connects s to the sinks. We say that s is the *root* of the single-source weighted spanner and the single-source weighted spanner is *rooted at s*. The definition for a single-sink weighted spanner where the terminal pairs share the same sink is defined similarly.

The online version of SINGLE-SOURCE WEIGHTED SPANNER is termed ONLINE SINGLE-SOURCE WEIGHTED SPANNER. For notation convenience, the vertex pair (s, t_i) denotes the i-th pair that arrives online.

Definition 1.8. ONLINE SINGLE-SOURCE WEIGHTED SPANNER

Instance: A directed graph $G = (V, E)$ with edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \text{poly}(n)\}$, and vertex pairs $D = \{(s, t_i) \mid t_i \in V, i \in [k]\}$ (k is unknown) with their corresponding pairwise distance bounds $Dist(s, t_i) \in \mathbb{Q}_{\geq 0}$ (where $Dist(s, t_i) \geq d_G(s, t_i)$) arrive online one at a time.

Objective: Upon the arrival of (s, t_i) with $Dist(s, t_i)$, construct a min-cost subgraph H of G such that $d_H(s, t_i) \leq Dist(s, t_i)$ by irrevocably selecting edges from E.

The state-of-the-art result for online directed Steiner trees is $\tilde{O}(k^\varepsilon)$-competitive implied by a more general online buy-at-bulk network design framework [14]. Our result is stated as follows and proved in Section 4.

Theorem 1.9. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized online algorithm for* ONLINE SINGLE-SOURCE WEIGHTED SPANNER *with approximation ratio $\tilde{O}(k^\varepsilon)$, which succeeds in resolving all pairs in D with high probability.*

Our online framework essentially generalizes the online Steiner forest problem by allowing distance constraints when edge lengths are positive integers of magnitude $\text{poly}(n)$. We note that the online algorithms also imply efficient algorithms for the corresponding offline problems with the same approximation ratios.

---

[3]Throughout the paper, the term *without distance constraints* means that the target distances are infinity. This is equivalent to the connectivity problem.

1.1.4 Summary

We summarize our main results for weighted spanners in Table 1 by listing the approximation (competitive) ratios and contrast them with the corresponding known approximation (competitive) ratios. We note that offline $\tilde{O}(k^{1/2+\varepsilon})$-approximate PAIRWISE WEIGHTED SPANNER and offline $\tilde{O}(k^{\varepsilon})$-approximate SINGLE-SOURCE WEIGHTED SPANNER can be obtained by our online algorithms.

Problem	Our Results	Previous Results
PAIRWISE WEIGHTED SPANNER	$\tilde{O}(n^{4/5+\varepsilon})$ (Thm 1.2) $\tilde{O}(k^{1/2+\varepsilon})$ (Thm 1.6)	$\tilde{O}(n^{4/5})$ (unit edge costs) [32] $\tilde{O}(n^{3/5+\varepsilon})$ (unit edge costs and lengths) [19] $\min\{\tilde{O}(k^{1/2+\varepsilon}), \tilde{O}(n^{2/3+\varepsilon})\}$ (without distance constraints) [10,18] $\tilde{O}(n^{4/7+\varepsilon})$ (unit edge costs and lengths, without distance constraints) [1]
ALL-PAIR WEIGHTED SPANNER	$\tilde{O}(n^{1/2+\varepsilon})$ (distance preservers, Thm 1.4)	$\tilde{O}(n^{1/2})$ (unit edge costs) [10] $3/2$ (without distance constraints) [54]
ONLINE PAIRWISE WEIGHTED SPANNER	$\tilde{O}(k^{1/2+\varepsilon})$ (Thm 1.6)	$\tilde{O}(n^{4/5})$ (unit edge costs) [32] $\min\{\tilde{O}(k^{1/2+\varepsilon}), \tilde{O}(n^{2/3+\varepsilon})\}$ (unit edge costs and lengths) [32] $\tilde{O}(k^{1/2+\varepsilon})$ (without distance constraints) [14]
SINGLE-SOURCE WEIGHTED SPANNER	$\tilde{O}(k^{\varepsilon})$ (also holds for online, Thm 1.9)	$O(k^{\varepsilon})$ (without distance constraints) [15] $\tilde{O}(k^{\varepsilon})$ (online, without distance constraints) [14]

Table 1: Summary of the approximation and competitive ratios. Here, n refers to the number of vertices and k refers to the number of terminal pairs. All edge lengths are positive integers in $\mathsf{poly}(n)$ and all edge costs are non-negative rational numbers. We note that PAIRWISE WEIGHTED SPANNER without distance constraints is equivalent to the directed Steiner forest problem. The all-pair weighted spanner problem without distance constraints is equivalent to the all-pair Steiner forest problem or the minimum strongly connected subgraph problem.

1.2 High-level technical overview

Most of the literature on approximation algorithms for *offline* spanner problems [10,11,19,21,28,32] partition the terminal pairs into two types: thin or thick. A pair $(s,t) \in D$ is *thin* if the graph $G^{s,t}$ induced by feasible $s \rightsquigarrow t$ paths has a small number of vertices, and *thick* otherwise. To connect each thick terminal pair (s,t), it is sufficient to sample vertices from the graph G to hit $G^{s,t}$, and then add shortest-path in-and-out-arborescences rooted at the sampled vertices. To connect each thin terminal pair (s,t), one uses a flow-based linear program (LP) and then rounds the solution.

1.2.1 Pairwise Weighted Spanners

For this problem, the goal is to approximately minimize the total cost while maintaining the required distances between terminal pairs, so it turns out that the approach for directed Steiner forests [10,28] is more amenable to this formulation. The approach for directed Steiner forests [10,28] is slightly different, namely, thick pairs are connected by adding cheap paths that contain at least one sampled vertex. In the

Steiner forest algorithms, there are usually three cases for the terminal pairs: 1) pairs that are thick and have low-cost connecting paths, 2) the majority of the remaining pairs have high-cost connecting paths, and 3) the majority of the remaining pairs have low-cost connecting paths.

With distance constraints, we have to modify the analysis for *all* three cases. The actual implementation of the strategy requires several new ideas and it significantly departs from the analysis of [10, 28] in several aspects, as we describe below.

In our first case, we cannot simply add cheap paths because they might violate the distance requirement. Instead, we add *feasible* cheap paths that satisfy the distance requirements, in order to connect the terminal pairs. For this purpose, we use the restricted shortest path FPTAS from [36, 45] as our subroutine.

The remaining two cases are resolved by using an iterative greedy algorithm based on a *density* argument. In each iteration, the greedy algorithm constructs a partial solution $E' \subseteq E$ with low density. We define the density of E' to be the ratio of the total edge cost of E' to the number of pairs connected by E'. Iteratively adding low-density partial solutions leads to a global solution of approximately minimum cost.

In the second case, [10, 28] use the low-density *junction trees* (the union of an in-arboresence and an out-arboresence rooted at the same vertex) from [18] in order to connect pairs with high-cost paths. However, the junction tree approximation in [18] cannot handle the distance constraints in our setting. Fortunately, with slight modifications, the junction tree approximation from [19] can be made to handle our distance requirements.

In the third case, [10, 28] formulate an LP where each edge has an indicator variable, then round the LP solution, and argue that with high probability, one can obtain a low-density partial solution that connects the terminal pairs with cheap paths. Two challenges arise in our setting. First, the LP formulation is different from the one in [10, 28] because we have to handle both distance and cost requirements. We resolve these constraints by using a different separation oracle from the previous literature [36, 45], namely we use the FPTAS for the *resource-constrained shortest path* problem from [38] (see Section 1.3.1 for more details). Secondly, in order to round the LP solution, we can no longer use the analysis in [10]. This is because the LP rounding scheme uses a union bound that depends on the number of the minimal subset of edges whose removal disconnects the terminal pairs (i.e., anti-spanners). Since we have to handle both lengths and costs in the LP constraints, we consider all possible subsets of edges and this is sufficient to achieve the $\tilde{O}(n^{4/5+\varepsilon})$-approximation.

1.2.2 All-pair Weighted Distance Preservers

For this problem, the solution takes advantage of the requirement that we have to *exactly* preserve the *all-pair* distances. It turns out that the strategy for the spanner problems [10, 11, 19, 21, 32] is more amenable. Recall that terminal pairs are partitioned into thin or thick.

To settle thick terminal pairs, most of the previous work that considers graphs with unit edge cost samples vertices from the graph G to hit $G^{s,t}$, and then adds shortest-path in-and-out-arborescences rooted at the sampled vertices. Note that the cost of an in-arborescence or an out-arborescence is always $n - 1$. However, with edge costs, it is not clear how the *cheapest* shortest-path in-and-out arborescences can be obtained. Instead, we add cheap *single-source and single-sink weighted distance preservers* rooted at the sampled vertices. This approach requires using the algorithm for ONLINE SINGLE-SOURCE WEIGHTED SPANNER described in details in Section 4. The key observation is that the terminal pairs of any single-source (single-sink) weighted distance preserver is a subset of all vertex pairs. This implies that any approximately optimal single-source (single-sink) weighted distance preserver must be cheap compared to the cost of the optimal all-pair weighted distance preserver.[4]

The approach that settles the thin pairs closely follows the algorithm in [10], which rounds a fractional solution of the LP for all-pair spanners. Different from PAIRWISE WEIGHTED SPANNER, we only have to handle the lengths in the LP constraints, so the analysis follows [10] and we can get a better approximation

---

[4]One might ask if the same approach for thick pairs works if we consider all-pair spanners instead of distance preservers. Unfortunately, for spanners, it is possible that a nice approximate solution connects some thick pairs by a cheap path that is feasible but not the shortest. Single-source and single-sink weighted distance preservers might be too expensive. This issue does not arise when edges have unit costs since the cost of any arborescence is always $n - 1$.

ratio. Ultimately, the costs for settling thick and thin pairs are both at most an $\tilde{O}(n^{1/2+\varepsilon})$ factor of the optimal solution.

1.2.3 Online Weighted Spanners

The main challenge for the online pairwise weighted spanner problem is that the standard greedy approach, which iteratively extracts low-density greedy solutions partially connecting terminal pairs, is no longer applicable. Another challenge is to handle the distance constraints for the terminal pairs that arrive online. Fortunately, the online spanner framework from [32] already adapts both the approach introduced in [14], which constructs a *collection of junction trees* in an online fashion, and the approach of [19], which judiciously handles distance constraints when edges have unit lengths. Our online results are obtained by extending the framework of [32] from graphs with unit edge costs to general edge costs.

1.3 Related work

1.3.1 Resource-constrained Shortest Path

In the resource-constrained shortest path problem [38] for directed networks, each edge is associated with r non-negative weights. Each type of weight $i \in [r-1]$ is associated with a budget. The r-th weight denotes the cost of the edge. The goal is to find a minimum-cost path that connects the single source to the single sink without violating the $r-1$ budgets. The results of [38] show that when r is a constant, there exists an FPTAS that finds a path with a cost at most the same as the feasible minimum-cost path by violating each budget by a factor of $1 + \varepsilon$. When $r = 2$, this problem is equivalent to the restricted shortest path problem [36, 45], which has been used extensively in the LP formulations for spanners and directed Steiner forests [10, 19, 21, 28, 32]. For our purpose, $r = 3$ because the LP formulation implicitly considers whether there exists a feasible path between terminal pairs whose cost exceeds a given threshold.

1.3.2 Undirected Bi-criteria Network Design

A general class of undirected bi-criteria network problems was introduced by [46]. A more related problem to ours is the undirected Steiner tree problem. The goal is to connect a subset of vertices to a specified root vertex. In the bi-criteria problem, the distance from the root to a target vertex is required to be at most the given global threshold. [46] presented a bi-criteria algorithm for undirected Steiner trees that is $O(\log n)$-approximate and violates the distance constraints by a factor of $O(\log n)$. Following [46], [35] extends this result to a more general buy-at-bulk bi-criteria network design problem, where the objective is polylog(n)-approximate and violates the distance constraints by a factor of polylog(n).

Recently, [29,34] studied the tree embedding technique used for undirected network connectivity problems with *hop* constraints. For a positively weighted graph with a global parameter $h \in [n]$, the *hop distance* between vertices u and v is the minimum weight among the u-v-paths using at most h edges. Under the assumption that the ratio between the maximum edge weight and the minimum edge weight is poly(n), the tree embedding technique allows a polylog(n)-approximation by relaxing the hop distance within a polylog(n) factor for a rich class of undirected network connectivity problems.

1.3.3 Other related directed network problems

The more related directed network problems are variants of spanners and Steiner problems, including directed Steiner trees [15, 56], directed Steiner network [18], fault-tolerance spanners [21, 22], and parameterized complexity analysis for directed s-spanners [31]. For a comprehensive account of the vast literature, we refer the reader to the excellent survey for spanners [2].

There is an extensive list of other related directed network problems, including distance preservers [12,19], approximate distance preservers [41], reachability preservers [1], and buy-at-bulk network design [5]. One direction along this line of research is to study the extremal bounds for the optimal subgraph in terms of the input parameters, instead of comparing the costs of an approximate and an optimal solution [1, 12, 41].

Another direction is to consider the online problem where terminal pairs arrive online and the goal is to irrevocably select edges so that the cost of the network is approximately minimized [3, 14, 32].

1.4 Organization

In Section 2, we present the $\tilde{O}(n^{4/5+\varepsilon})$-approximation algorithm for PAIRWISE WEIGHTED SPANNER. In Section 3, we present the $\tilde{O}(n^{1/2+\varepsilon})$-approximation algorithm for ALL-PAIR WEIGHTED DISTANCE PRE-SERVER. In Section 4, we present the $\tilde{O}(k^{1/2+\varepsilon})$-competitive algorithm for ONLINE PAIRWISE WEIGHTED SPANNER and the $\tilde{O}(k^{\varepsilon})$-competitive algorithm for ONLINE SINGLE-SOURCE WEIGHTED SPANNER.

2 Pairwise Weighted Spanners

We recall the definition of the PAIRWISE WEIGHTED SPANNER problem.

Definition 1.1. PAIRWISE WEIGHTED SPANNER
  Instance: A directed graph $G = (V, E)$ with n vertices and edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \mathsf{poly}(n)\}$, and a set $D \subseteq V \times V$ of vertex pairs and their corresponding pairwise distance bounds $Dist : D \to \mathbb{Q}_{\geq 0}$ (where $Dist(s, t) \geq d_G(s, t)$) for every terminal pair $(s, t) \in D$.
  Objective: Find a min-cost subgraph H of G such that $d_H(s, t) \leq Dist(s, t)$ for all $(s, t) \in D$.

In this section, we prove Theorem 1.2.

Theorem 1.2. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized algorithm for* PAIRWISE WEIGHTED SPANNER *with approximation ratio $\tilde{O}(n^{4/5+\varepsilon})$, which succeeds in resolving all pairs in D with high probability.*

For ease of presentation, we assume that we have a guess for the cost of the optimal solution - OPT for that instance as in [10, 28]. Let τ denote the value of our guess. We set $\tau_0 = \min_{e \in E}\{c(e) \mid c(e) > 0\}$; then we carry out multiple iterations of our overall procedure setting τ to be equal to an element in $\{(\tau_0, 2 \cdot \tau_0, 4 \cdot \tau_0, \ldots, 2^i \cdot \tau, \ldots, \sum_{e \in E} c(e))\}$ for those iterations. Finally, we take the cheapest spanner from across all these iterations. Thus, it is sufficient to give the approximation guarantee for the iteration when OPT $\leq \tau \leq 2 \cdot$ OPT. We can obtain this guess in $O(\log(\sum_{e \in E} c(e)/\tau_0))$ iterations, which is polynomial in the input size.

We next define some notions that are commonly used in the spanner and Steiner forest literature. Let us fix some useful parameters $\beta = n^{3/5}$, and $L = \tau/n^{4/5}$. We say that a path $p(s, t)$ that connects a specific terminal pair (s, t) is *feasible* if $\sum_{e \in p(s,t)} \ell(e) \leq Dist(s, t)$. We say that $p(s, t)$ is *cheap* if the $\sum_{e \in p(s,t)} c(e) \leq L$. We say that a terminal pair $(s, t) \in D$ is *thick* if the *local graph* $G^{s,t} = (V^{s,t}, E^{s,t})$ induced by the vertices on feasible $s \rightsquigarrow t$ paths of cost at most L has at least n/β vertices; we say it is *thin* otherwise. We note that the definitions of thick and thin pairs are slightly different from how they are defined in [10, 28] (see Remark 2.4 for the precise reason). We say that a set $E' \subseteq E$ settles (or resolves) a pair $(s, t) \in D$ if the subgraph (V, E') contains a feasible $s \rightsquigarrow t$ path.

2.1 Resolving thick pairs

Let $S = \{s \mid \exists t : (s, t) \in D\}$ and $T = \{t \mid \exists s : (s, t) \in D\}$. We first settle the thick pairs with high probability. We do this by sampling some vertices using Algorithm 1 and then adding some incoming paths and outgoing paths from the samples to the vertices in S and T respectively using Algorithm 2. We have to ensure that any path we build is both feasible and cheap and we do that with the help of a black box for the RESTRICTED SHORTEST PATH PROBLEM (see Definition 2.2) which we formally define later.

Claim 2.1. *Algorithm 1 selects a set of samples R such that with high probability any given thick pair (s, t) has at least one vertex from its local graph in R.*

Proof. The standard proof is deferred to Appendix A.1. □

Algorithm 1 Sample($G(V, E)$)

1: $R \leftarrow \phi$, $k \leftarrow 3\beta \ln n$.
2: Sample k vertices independently and uniformly at random and store them in the set R.
3: **return** R.

In Algorithm 2, we call Algorithm 1 to get a set of samples R. Then, for each $u \in R, s \in S, t \in T$, we try to add a shortest $s \rightsquigarrow u$ path and a shortest $u \rightsquigarrow v$ path each of cost at most L, of increasing length.

For this purpose, we use the restricted shortest path (bi-criteria path) problem from [45].

Definition 2.2. RESTRICTED SHORTEST PATH

Instance: A directed graph $G = (V, E)$, edge lengths $\ell : E \rightarrow \mathbb{R}_{\geq 0}$, edge costs $c : E \rightarrow \mathbb{R}_{\geq 0}$, a vertex pair $(s, t) \in V \times V$, and a threshold $T \in \mathbb{R}_{>0}$.

Objective: Find a minimum cost $s \rightsquigarrow t$ path P such that $\sum_{e \in P} \ell(e) \leq T$.

The following lemma from [36, 45] gives an FPTAS for RESTRICTED SHORTEST PATH.

Lemma 2.3. ([36, 45]) *There exists an FPTAS for* RESTRICTED SHORTEST PATH *that gives a* $(1 + \varepsilon, 1)$ *approximation, i.e., the path ensures that* $\sum_{e \in P} \ell(e) \leq T$ *and has a cost at most* $1 + \varepsilon$ *times the optimal.*

Using Lemma 2.3 as our black box, we binary search for a path of length between $\min_{e \in E}\{\ell(e)\}$ and $n \cdot \max_{e \in E}\{\ell(e)\}$ that will give us a cheap $s \rightsquigarrow u$ path. Since the edge lengths and thus the path lengths are all integers, this is possible in $O(\log(n \cdot \max_{e \in E}\{\ell(e)\}))$ iterations which is polynomial in the input size. It is possible that we never find an $s \rightsquigarrow u$ path of cost less than L, in which case we just ignore this (s, u) pair. We then do the same for all the (u, t) pairs. See the full details in Algorithm 2.

Remark 2.4. The need to use Lemma 2.3, as opposed to a simple shortest path algorithm, is the main difference we have from [10] for resolving thick pairs. We have to ensure that any path we add is both cheap and feasible, unlike [10], which just needs to ensure that the path is cheap. Note that this is the reason our definition of thick pairs differs from that of [10] which necessitates some more major modifications in subsequent parts of the algorithm as well.

Algorithm 2 Thick pairs resolver $(G(V, E), \{\ell(e), c(e)\}_{e \in E})$

1: $R \leftarrow \phi$, $G' \leftarrow \phi$.
2: $R \leftarrow$ Sample($G(V, E)$).
3: **for** $u \in R$ **do**
4:      **for** $s \in S$ **do**
5:          Use RSP to binary search for the shortest $s \rightsquigarrow u$ path of cost at most $L \cdot (1 + \varepsilon)$ and add it to G' if it exists.
6: **for** $u \in R$ **do**
7:      **for** $t \in T$ **do**
8:          Use RSP to binary search for the shortest $u \rightsquigarrow t$ path of cost at most $L \cdot (1 + \varepsilon)$ and add it to G' if it exists.
9: **return** G'

Lemma 2.5. *With high probability, the set of edges returned by Algorithm 2 resolves all thick pairs in D with a total cost $\tilde{O}(n^{4/5} \cdot \tau)$. Moreover, Algorithm 2 runs in polynomial time.*

Proof. If some $u \in R$ was originally in the local graph $G^{s,t}$, then Algorithm 2 would have added at least one $s \rightsquigarrow u \rightsquigarrow t$ path from $G^{s,t}$ that is feasible and has cost less than $2L(1 + \varepsilon)$. This is because if u was in the local graph of (s, t), then there exists an $s \rightsquigarrow u$ path of cost less than L of some length $\ell \in [n \cdot \max_{e \in E}\{\ell(e)\}]$. Since we binary search over the possible values for ℓ and take the lowest possible one, we will find such a

9

path with distance at most the minimum length of an $s \rightsquigarrow t$ path that is cheap. Note that we could have a smaller distance because we use a larger bound for cost for our path in comparison to the local graph. Using Claim 2.1, such a cheap and feasible path will exist with a high probability for all $(s,t) \in D$ that are thick for some samples $u \in R$.

Now we analyze the cost of Algorithm 2. The cost from all the edges we add in Algorithm 2 would be $O(n \cdot k \cdot L)$. This is because we pick k samples and each of them needs to add an incoming and outgoing path of cost $L(1+\varepsilon)$ to at most n vertices. Plugging in the values for k and L, we can see that the total cost would be $\tilde{O}(n^{4/5} \cdot \tau)$. In addition, note that Algorithm 2 will run in polynomial time because our binary search only needs to search the integers in $[\min_{e \in E}\{\ell(e)\}, n \cdot \max_{e \in E}\{\ell(e)\}]$.[5] $\qquad\square$

2.2 Resolving thin pairs

Now we focus on the thin pairs after removing the settled thick pairs from the set D. We define the *density* of a set of edges E' to be the ratio of the total cost of these edges to the number of pairs settled by those edges. We first describe how to efficiently construct a subset K of edges with density $\tilde{O}(n^{4/5+\varepsilon})\tau/|D|$. Then we iteratively find edge sets with that density, remove the pairs, and repeat until we resolve all thin pairs. This gives a total cost of $\tilde{O}(n^{4/5+\varepsilon} \cdot \tau)$.

We construct K by building two other sets K_1 and K_2 and picking the smaller density of them. Let H be an optimal solution with cost τ. Let C be the set of demand pairs for which the minimum cost of a feasible $s \rightsquigarrow t$ path in H is at least L (note that the local graph for these pairs would be empty).

We have two cases:

- $|D|/2 \leq |C| \leq |D|$ and

- $0 \leq |C| < |D|/2$.

2.2.1 When $|D|/2 \leq |C| \leq |D|$

We will use the notion of *junction tree* as a black box for resolving this case. Informally, junction trees are trees that satisfy significant demand at low cost. They have a root node r, a collection of paths into r, and paths out of r to satisfy some of this demand. In our case, we also need these paths to be cheap and short. The following is a formal definition of a junction tree variant that fits the needs of our problem.

Definition 2.6. *Distance-preserving Weighted Junction Tree*

Let $G = (V, E)$ be a directed graph with edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$, edge costs $c : E \to \mathbb{R}_{\geq 0}$, and a set $D \subseteq V \times V$ of ordered pairs and their corresponding pairwise distance bounds $Dist : D \to \mathbb{R}$ (where $Dist(s,t) \geq d_G(s,t)$ for every terminal pair $(s,t) \in D$), and a root $r \in V$. We define *distance-preserving weighted junction tree* to be a subgraph H of G that is a union of an in-arboresences and an out-arboresences both rooted at r containing an $s \rightsquigarrow t$ path going through the root r of length at most $Dist(s,t)$, for one or more $(s,t) \in D$.

The *density* of a junction tree is defined as the ratio of the sum of costs of all edges in the junction tree to the number of pairs settled by the junction tree.

Claim 2.7. *If $|D|/2 \leq |C| \leq |D|$, then there exists a distance-preserving weighted junction tree of density* $O\left(n^{4/5} \cdot \tau/|D|\right)$.

Proof. Let H be an optimal solution subgraph of G that connects all the costly thin pairs. Take the paths in H connecting the pairs in C. The sum of the costs of all such paths is at least $|C|L$. Now, let μ be the maximum number of these paths that any edge in G belongs to. The sum of the costs of the paths is at most $\mu \cdot \tau$ and thus there must exist an edge belonging to $\mu \geq |C|L/\tau$ paths. Pick such an arbitrary edge and call it the *heavy-enough edge*, and call its source as the *heavy-enough vertex*, denoted h_v. Now, consider

---

[5]Note that Algorithm 2 will still run in polynomial-time if we use an exhaustive search instead of a binary search since the edge lengths are in $\mathsf{poly}(n)$.

a tree made by adding feasible paths from $s \in S$ to h_v and h_v to $t \in T$ that satisfies at least μ pairs. We do not add an edge if it is not in H. This ensures that the cost of this tree is less than τ. This tree would connect at least μ pairs, and thus it would have a density at most $\tau/\mu = \tau^2/(|C|L)$.

If $L = \tau/n^{4/5}$, then $\tau^2/(|C|L) = n^{4/5} \cdot \tau/|C|$. If $|D| > |C| > |D|/2$, we have $n^{4/5} \cdot \tau/|C| = O\left(n^{4/5} \cdot \tau/|D|\right)$ and thus we have proved the existence of a junction tree of the required density. $\qquad\square$

Remark 2.8. The analysis so far for this case is quite similar to [10]. The counting argument to establish the existence of a *heavy-enough edge* is identical. However, the usage of this counting argument to get a structure of the required density is slightly different because we need to account for feasibility. The main difference however is the fact that we need to use distance-preserving weighted junction tree as opposed to regular junction trees.

The following lemma is essentially from [19] (Theorem 5.1). But the small yet important modifications that we need are not covered in [19]. We defer the proof to Appendix A.2.

Lemma 2.9. *For any constant $\varepsilon > 0$, there is a polynomial-time approximation algorithm for the minimum density distance-preserving junction tree as long as the edge lengths are integral and polynomial in n. In other words, there is a polynomial time algorithm which, given a weighted directed n vertex graph $G = (V, E)$ where each edge $e \in E$ has a cost $c(e) \in \mathbb{R}_{\geq 0}$ and integral length $\ell(e) \in \{1, 2, ..., \mathsf{poly}(n)\}$, terminal pairs $P \subseteq V \times V$, and distance bounds $Dist : P \to \mathbb{N}$ (where $Dist(s, t) \geq d_G(s, t)$) for every terminal pair $(s, t) \in P$, approximates the following problem to within an $O(n^\varepsilon)$ factor:*

- *Find a non-empty set of edges $F \subseteq E$ minimizing the ratio:*

$$\min_{r \in V} \frac{\sum_{e \in F} c(e)}{|\{(s, t) \in P | d_{F,r}(s, t) \leq Dist(s, t)\}|} \qquad (1)$$

where $d_{F,r}(s, t)$ is the length of the shortest path using edges in F which connects s to t while going through r (if such a path exists).

Lemma 2.10. *When $|D|/2 \leq |C| \leq |D|$, we can get a set of edges K_1 that has density at most $\tilde{O}(n^{4/5+\varepsilon} \cdot \tau/|D|)$.*

Proof. From Claim 2.7, we can see that there exists a distance-preserving weighted junction tree of density at most $O(n^{4/5} \cdot \tau/|D|)$. Now, we can just use Lemma 2.9 to get a distance-preserving weighted junction tree with density at most $\tilde{O}(n^{4/5+\varepsilon} \cdot \tau/|D|)$ and store the edges returned by it in K_1. $\qquad\square$

2.2.2 When $0 \leq |C| < |D|/2$

To handle this case, we build a linear program (LP) that fits our problem's requirements, solve it approximately with the help of a separation oracle, and finally round it to get a set of edges with density $\tilde{O}(n^{4/5+\varepsilon}) \cdot \tau/|D|$. The linear program is quite similar to the one used in [10, 28], but it has a subtle distinction that significantly changes the tools and proof techniques we have to use. We will be referring to [28] quite frequently in this section because [10] does not directly present a way to solve the LP (it relies on [28] for this).

Building and solving the linear program We will need the following definition in order to set up a relevant LP. For $(s, t) \in D$, let $\Pi(s, t)$ be the set of all *feasible* $s \rightsquigarrow t$ paths of cost at most L, and let $\Pi = \cup_{(s,t) \in D} \Pi(s, t)$. Each edge e has a capacity x_e, each path $p \in \Pi$ carries f_p units of flow, and $y_{s,t}$ is the total flow through all paths from s to t. Define a linear program as follows:

$$\min \qquad \sum_{e \in E} c(e) \cdot x_e$$

$$\text{subject to} \quad \sum_{(s,t) \in D} y_{s,t} \geq \frac{|D|}{2},$$

$$\sum_{\Pi(s,t) \ni p \ni e} f_p \leq x_e \qquad \forall (s,t) \in D, e \in E, \qquad (2)$$

$$\sum_{p \in \Pi(s,t)} f_p = y_{s,t} \qquad \forall (s,t) \in D,$$

$$0 \leq y_{s,t}, f_p, x_e \leq 1 \quad \forall (s,t) \in D, p \in \Pi, e \in E.$$

LP (2) tries to connect at least $|D|/2$ pairs from D using paths of cost at most L while minimizing the total cost of the edges that are used. It is almost identical to the corresponding LP in [10, 28] for Steiner forests, except that we consider only *feasible* paths that are cheaper than L, while they consider all paths that are cheaper than L.

Lemma 2.11. *Let* OPT *be the optimal value of an instance of* Pairwise Weighted Spanner. *Then, the optimal value of LP* (2) *corresponding to that instance is at most* OPT. *In addition, a solution for LP* (2) *of value at most* $(1 + \varepsilon) \cdot$ OPT *can be found in polynomial time.*

Proof. Let us consider the dual of LP (2).

$$\max \qquad \sum_{e \in E} x_e + \sum_{(s,t) \in D} y_{s,t} - W \cdot \frac{|D|}{2} \qquad (3a)$$

$$\text{subject to} \qquad \sum_{(s,t) \in D} z_{(s,t),e} + c(e) \leq x_e \qquad \forall e \in E, \qquad (3b)$$

$$y_{s,t} + w_{s,t} \geq W \qquad \forall (s,t) \in D, \qquad (3c)$$

$$w_{s,t} \leq \sum_{e \in p} z_{(s,t),e} \qquad \forall (s,t) \in D, p \in \Pi(s,t), \qquad (3d)$$

$$W, x_e, y_{s,t}, z_{(s,t),e} \geq 0 \qquad \forall (s,t) \in D, e \in E. \qquad (3e)$$

Our dual is slightly different from the dual in [10, 28]. Constraints in (3b), (3c), and (3e) are identical, but constraints in (3d) are slightly different because the set of paths Π we consider are different from the set of paths considered in [28]. As in [28], we could find violating constraints for those constraints in (3b), (3c), and (3e) in polynomial time. The only constraints that require care are the constraints in (3d), which may be exponentially many.

When we consider a single (s,t) pair, [28] pointed out that their variant of constraints in (3d) are equivalent to Restricted Shortest Path which is NP-hard, see [36, 45]. [28] then uses an FPTAS [36, 45] for Restricted Shortest Path as an approximate separation oracle for those constraints. But we need a different separation oracle because the set of paths Π allowed in our LP have two restrictions (as opposed to [28] which has only one) in addition to an objective. We now define the Resource-constrained Shortest Path problem that is presented in [38].

Definition 2.12. Resource-constrained Shortest Path (k-RCSP)
 Instance: A directed graph $G = (V, E)$, with edge costs $c : E \to \mathbb{Q}_{\geq 0}$, and a pair (s, t). For each edge $e \in E$, we have a vector $r_e = (r_{1,e}, r_{2,e}, \ldots, r_{k,e})$ of size k where each $r_{i,e} \in \mathbb{Q}_{\geq 0} \, \forall i \in [k]$.
 Objective: Find a minimum cost $s \rightsquigarrow t$ path P such that $\sum_{e \in P} r_{i,e} \leq R_i, \, \forall i \in [k]$.

Claim 2.13. *2-RCSP acts as a separation oracle for those constraints in equation* (3d) *that correspond to a specific* $(s,t) \in D$.

Proof. We can use one of the resource constraints in 2-RCSP for ensuring that the distance constraints for (s,t) are satisfied and use the other resource constraint to ensure that $w_{s,t} > \sum_{e \in P} z_{(s,t),e}$. In other words, we use one resource to model the edge lengths and another to model the dual variable $z_{\{s,t\},e}$. We can now try to find a minimum cost $s \rightsquigarrow t$ path in this instance of 2-RCSP where costs for 2-RCSP are equivalent to the costs in our instance of Pairwise Weighted Spanner. If the minimum cost obtained when we meet these constraints is less than L, then we have a violating constraint and if not we do not have one. □

The Resource-constrained shortest path problem is NP-hard [38]. So, we instead get a separation oracle for an approximate variant of LP (3). Now, given resource constraints $R_1, R_2, \ldots, R_k$ for the Resource-constrained shortest path problem, let OPT_{RCSP} be the cost of the minimum cost $s \rightsquigarrow t$ path that satisfies the resource constraints. An $(1; 1 + \varepsilon, \ldots, 1 + \varepsilon)$-approximation scheme finds an $s \rightsquigarrow t$ path whose cost is at most OPT_{RCSP}, but the resource constraints are satisfied up to a factor of $1 + \varepsilon$ for that path.

Lemma 2.14. (RCSP- [38]) *If k is a constant then there exists a fully polynomial time $(1; 1+\varepsilon, \ldots, 1+\varepsilon)$-approximation scheme for the k-RCSP that runs in time polynomial in input size and $1/\varepsilon$.*

We have to be careful in our usage of Lemma 2.14. The FPTAS for the Restricted Shortest Path problem from [45] cleanly serves the requirements of [28] as it is a $(1 + \varepsilon; 1)$ approximation and it can strictly satisfy the constraints. [28] then uses these constraints to ensure that the weight requirements are strictly met. But the FPTAS given by [38] does not strictly satisfy the constraints since we need both the length and weight constraints to be satisfied strictly.

We overcome this obstacle by re-purposing the objective to handle the edge weights and by carefully ensuring that any error in the path length caused by using the $1 + \varepsilon$ approximation from [38] does not make us use an incorrect path. To ensure that we do not select an incorrect path, it is sufficient to ensure that the potential error from [38] is less than any error that is possible in our given input graph. Since the edge lengths are positive integers, observe that for any two $s \rightsquigarrow t$ paths with different lengths, the length difference is at least one. In addition, the path lengths are at most $n \cdot \max_{e \in E}\{\ell(e)\}$. Since all edge lengths are integral and of magnitude $\mathsf{poly}(n)$, it is sufficient to have $\varepsilon \leq 1/(n \cdot \max_{e \in E}\{\ell(e)\})$. Thus, we can fix ε such that $1/\varepsilon = O(n \cdot \mathsf{poly}(n))$ to ensure that the running time will remain polynomial in input size and strictly satisfy the distance constraints.

Now, we take an approximate version of LP (3) which is the following LP

$$
\begin{aligned}
\max \quad & \sum_{e \in E} x_e + \sum_{s,t} y_{s,t} - W \cdot \frac{|D|}{2} \\
\text{subject to} \quad & \sum_{s,t} z_{(s,t),e} + c(e) \leq x_e && \forall e \in E, \\
& y_{s,t} + w_{s,t} \geq W && \forall (s,t) \in D, \\
& (1 + \varepsilon) \cdot w_{s,t} \leq \sum_{e \in p} z_{(s,t),e} && \forall (s,t) \in D, p \in \Pi(s,t), \\
& W, x_e, y_{s,t}, z_{(s,t),e} \geq 0 && \forall (s,t) \in D, e \in E.
\end{aligned}
\tag{4}
$$

We can exactly solve LP (4) using [38] and thus we can also exactly solve the dual of LP (4) which would be:

$$\min \qquad \sum_{e \in E} c(e) \cdot x_e$$

$$\text{subject to} \qquad \sum_{(s,t) \in D} y_{s,t} \geq \frac{|D|}{2},$$

$$\sum_{\Pi(s,t) \ni P \ni e} f_p \leq x_e \cdot (1 + \varepsilon) \qquad \forall (s,t) \in D, e \in E, \qquad (5)$$

$$\sum_{P \in \Pi(s,t)} f_p = y_{s,t} \qquad \forall (s,t) \in D,$$

$$0 \leq y_{s,t}, f_p, x_e \leq 1 \qquad \forall (s,t) \in D, p \in \Pi, e \in E.$$

Let $\mathsf{OPT}(\varepsilon)$ and OPT be the optimal values to (5) and (2), respectively. Observe that $\mathsf{OPT}(\varepsilon) \leq \mathsf{OPT}$ because the constraints in LP (5) are slacker than the constraints in (2) and both these LPs are minimization LPs. Also note that if $\hat{x}(\varepsilon)$ is a feasible solution to (5), then by replacing the value of every variable x_e in $\hat{x}(\varepsilon)$ by $\min\{1, x_e \cdot (1 + \varepsilon)\}$, we get a new solution $\hat{x}$ which is a feasible solution to (2). The value of the optimal solution then is at most $(1 + \varepsilon) \cdot \mathsf{OPT}(\varepsilon) \leq (1 + \varepsilon) \cdot \mathsf{OPT}$. $\qquad \square$

Rounding our solution Now we need to round the solution of LP (2) appropriately to decide which edges we need to include in our final solution. The overall structure of our rounding procedure is similar to that of [10], but there are some important differences in the proof techniques we use here because the nature of our problem prevents us from using some of the techniques used by [10]. Let $\{\hat{x}_e\} \cup \{\hat{y}_{s,t}\}$ be a feasible approximate solution to LP (2). Let K_2 be the set of edges obtained by running Algorithm 3 on $\{\hat{x}_e\}$.

Algorithm 3 Thin pair rounding [LP rounding] (x_e)

1: $E'' \leftarrow \phi$.
2: **for** $e \in E$ **do**
3:      Add e to K_2 with probability $\min\{n^{4/5} \ln n \cdot x_e, 1\}$;
4: **return** E''

The following lemma is an adaptation of Claim 2.3 from [10].

Claim 2.15. *Let $A \subseteq E$. If Algorithm 3 receives a fractional vector $\{\hat{x}_e\}$ with nonnegative entries satisfying $\sum_{e \in A} \hat{x}_e \geq 2/5$, the probability that it outputs a set E'' disjoint from A is at most $\exp(-2n^{4/5} \cdot \ln n/5)$.*

Proof. If A contains an edge e which has $\hat{x}_e \geq 1/(n^{4/5} \ln n)$, then e is definitely included in E''.

Otherwise, the probability that no edge in A is included in E'' is

$$\prod_{e \in A} (1 - n^{4/5} \ln n \cdot \hat{x}_e) \leq \exp\left(-\sum_{e \in A} n^{4/5} \ln n \cdot \hat{x}_e\right) \leq \exp\left(-\frac{2}{5} n^{4/5} \ln n\right).$$

$\square$

Let us now define anti-spanners which serve as a useful tool to analyze the rounding algorithm for our LP. Our definition of anti-spanners is slightly different from Definition 2.4 in [10] to account for the fact we also have distance constraints.

Definition 2.16. A set $A \subseteq E$ is an anti-spanner for a terminal pair $(s,t) \in E$ if $(V, E \setminus A)$ contains no feasible path from s to t of cost at most L. If no proper subset of anti-spanner A for (s,t) is an anti-spanner for (s,t), then A is minimal. The set of all minimal anti-spanners for all thin edges is denoted by $\mathcal{A}$.

The following lemma is an analogue of Claim 2.5 from [10].

14

Lemma 2.17. *Let $\mathcal{A}$ be the set of all minimal anti-spanners for thin pairs. Then $|\mathcal{A}|$ is upper-bounded by $|D| \cdot 2^{(n/\beta)^2/2}$.*

Proof. Let $PS(s,t)$ be the power set of all edges in the local graph for a given thin pair (s,t). Since (s,t) is a thin pair we have at most n/β vertices and $(n/\beta)^2/2$ edges in the local graph, therefore $|PS(s,t)| \leq 2^{(n/\beta)^2/2}$ for any (s,t) that is a thin pair. Now, every anti-spanner for a specific demand pair $(s,t) \in D$ is a set of edges and therefore corresponds to an element in $PS(s,t)$. Let $PS_{\text{thin}} = \bigcup_{(s,t)} PS(s,t)$ where $(s,t) \in D$ are thin pairs. Every anti-spanner for a thin pair is a set of edges and therefore corresponds to an element in PS_{thin}. We have $|\mathcal{A}| \leq |PS_{\text{thin}}| \leq |D| \cdot 2^{(n/\beta)^2/2}$ which proves the lemma. $\square$

Remark 2.18. Our upper bound for the $|\mathcal{A}|$ is different from the upper bound in [10] because we cannot use the techniques used by [10] to give a bound for $|\mathcal{A}|$. They use some arborescences which have a one-to-one correspondence with the set of anti-spanners and it is hard to get an equivalent structure because we have both lengths and weights on each edge. This step is responsible for the fact that we do not have the $\tilde{O}(n^{2/3+\varepsilon})$-approximation as in [10].

The rest of this discussion is quite similar to [10] although the exact constants and the expressions involved are different because of the result in Lemma 2.17. Lemma 2.19 is similar to Lemma 5.2 from [10].

Lemma 2.19. *With high probability set K_2 settles every thin pair (s,t) with $\hat{y}_{s,t} \geq 2/5$.*

Proof. For every thin pair $(s,t) \in D$ with $\hat{y}_{s,t} \geq 2/5$, if A is an anti-spanner for (s,t) then $\sum_{e \in A} \hat{x}_e \geq \sum_{P \in \Pi(s,t)} \hat{f}_p \geq 2/5$, where $\hat{f}_p$ is the value of the variable f_p in LP (2) that corresponds to the solution $\{\hat{x}_e\} \cup \{\hat{y}_{s,t}\}$.

By Claim 2.15, the probability that A is disjoint from K_2 is at most $\exp(-2n^{4/5} \cdot \ln n/5)$. Further using Lemma 2.17, we can bound the number of minimal anti-spanners for thin pairs and then if we apply union bound, we have the probability that K_2 is disjoint from any anti spanner for a thin pair is at most

$$\exp\left(-\frac{2}{5}n^{4/5} \cdot \ln n\right) \cdot |D| \cdot 2^{(n/\beta)^2/2}. \tag{6}$$

In the worst case, $|D|$ is n^2. Recall that $\beta = n^{3/5}$, we have $(n/\beta)^2 = n^{4/5}$ and thus (6) becomes

$$\exp\left(-\frac{2}{5} \cdot n^{4/5} \cdot \ln n + \ln\left(n^2 \cdot 2^{n^{4/5}/2}\right)\right) = \exp\left(-\Theta(n^{4/5} \ln n)\right).$$

Thus we have shown that the probability K_2 is disjoint from any anti-spanner for a thin pair is exponentially small when $\hat{y}_{s,t} \geq 2/5$. $\square$

Lemma 2.20. *When $0 \leq |C| < |D|/2$, with high probability, the density of K_2 is at most*

$$\tilde{O}(n^{4/5} \cdot \tau/|D|).$$

Proof. Firstly notice that the expected cost of K_2 would be at most $n^{4/5} \ln n \cdot \tau$. We also point out that the number of pairs $(s,t) \in D$ for which $\hat{y}_{s,t} < 2/5$ is at most $5|D|/6$ because otherwise the amount of flow between all pairs is strictly less than $|D|/2$ which violates a constraint of LP (2). Since with high probability all pairs for which $\hat{y}_{s,t} \geq 2/5$ are satisfied, this means that the expected density of K_2 is at most

$$\frac{n^{4/5} \ln n \cdot \tau}{|D|/6} = \frac{6n^{4/5} \ln n \cdot \tau}{|D|} = \frac{\tilde{O}(n^{4/5} \cdot \tau)}{|D|}.$$

$\square$

Now we are ready to prove Theorem 1.2.

Proof of Theorem 1.2. Using Lemma 2.5 we can resolve all thick pairs with high probability with cost at most $\tilde{O}(n^{4/5+\varepsilon})$. Then, we can make two sets of edges K_1 and K_2 using a distance-preserving weighted junction tree and by rounding the approximate solution to LP (2) respectively. By Lemmas 2.10 and 2.20, we can see that at least one of them will have a density at most $\tilde{O}(n^{4/5} \cdot \tau/|D|)$. If we take the cheaper among them and keep iterating we can resolve all thin pairs with a high probability and with cost at most $\tilde{O}(n^{4/5+\varepsilon})$. $\square$

3   All-pair Weighted Distance Preservers

We recall the definition of the ALL-PAIR WEIGHTED DISTANCE PRESERVER problem.

Definition 1.3. ALL-PAIR WEIGHTED DISTANCE PRESERVER
    Instance: A directed graph $G = (V, E)$ with edge costs $c : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \{1, 2, 3, ..., \mathsf{poly}(n)\}$.
    Objective: Find a min-cost subgraph H of G such that $d_H(s, t) = d_G(s, t)$, for all $(s, t) \in V \times V$.

    In this section, we prove Theorem 1.4.

Theorem 1.4. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized algorithm for* ALL-PAIR WEIGHTED DISTANCE PRESERVER *with approximation ratio $\tilde{O}(n^{1/2+\varepsilon})$, which succeeds in resolving all pairs in $V \times V$ with high probability.*

    Our proof structure for this subsection is very similar to that of [10] except for our use of single sink and single source spanners.
    As in Section 2, we assume that we have a guess for the cost of the optimal solution - OPT for the given instance of ALL-PAIR WEIGHTED DISTANCE PRESERVER. Let τ denote the value of our guess. Let us set $\beta = n^{1/2}$. We say that a terminal pair $(s, t) \in D$ is *thick* if the *local graph* $G^{s,t} = (V^{s,t}, E^{s,t})$ induced by the vertices on feasible paths from s to t has at least n/β vertices; we say it is *thin* otherwise. We note that the definitions of thick and thin pairs are slightly different from how they are defined in Section 2 as we only care about the feasibility of a path, not its cost. We say that a set $E' \subseteq E$ settles (or resolves) a pair $(s, t) \in D$ if the subgraph (V, E') contains a feasible path from s to t.

3.1   Thick pairs

We first resolve the thick pairs by randomly sampling vertices and building single-source and single-sink spanners from the samples using Theorem 1.9. We then resolve thin pairs by building a linear program, solving it, and rounding as in [10]. As mentioned earlier, our definition of thick and thin pairs is different in this section when compared to Section 2, and this allows us to use a much simpler proof (although one that will be effective only in the case of weighted distance preservers as opposed to the more general weigthed spanners).

Algorithm 4 Thick pairs resolver - Distance preserver $(G(V, E), \{\ell(e), c(e)\}_{e \in E})$

1: $R \leftarrow \phi$, $G' \leftarrow \phi$.
2: **for** $i = 1$ to $\beta \ln n$ **do**
3:     $v \leftarrow$ a uniformly random element of V.
4:     $S_v^{source} \leftarrow$ a single-source distance preserver rooted at v with $D = \{v\} \times V$ by using Theorem 1.9.
5:     $S_v^{sink} \leftarrow$ a single-sink distance preserver rooted v with $D = \{v\} \times V$ by using Theorem 1.9.
6:     $G' \leftarrow G' \cup S_v^{source} \cup S_v^{sink}$, $R \leftarrow R \cup \{v\}$.
7: **return** G'

Lemma 3.1. *Algorithm 4 resolves all thick pairs for* ALL-PAIR WEIGHTED DISTANCE PRESERVER *with high probability and cost $\tilde{O}(n^\varepsilon \cdot \beta \cdot \textsf{OPT}) = \tilde{O}(n^{1/2+\varepsilon} \cdot \textsf{OPT})$.*

Proof. Let $\mathsf{OPT}(S_v^{source})$ be the optimal costs of a single-source distance preserver rooted at v with $D = \{v\} \times V$ and $\mathsf{OPT}(S_v^{sink})$ be the optimal costs of a single-sink distance preserver rooted at v with $D = \{v\} \times V$.

We recall Theorem 1.9.

Theorem 1.9. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized online algorithm for* ONLINE SINGLE-SOURCE WEIGHTED SPANNER *with approximation ratio $\tilde{O}(k^\varepsilon)$, which succeeds in resolving all pairs in D with high probability.*

This theorem also gives an $\tilde{O}(k^\delta)$-approximation for the offline problem SINGLE-SOURCE WEIGHTED SPANNER for any constant $\delta > 0$. Single-sink distance preservers can be obtained by simply reversing the edges. The number of terminal pairs $k = \Theta(n^2)$. By setting $\delta = \varepsilon/2$ and the target distances to the exact distances in G for all vertex pairs, we observe that the cost due to one sample in Algorithm 4 is at most $\tilde{O}(n^\varepsilon(\mathsf{OPT}(S_v^{source}) + \mathsf{OPT}(S_v^{sink})))$. Note that a distance preserver for all pairs also serves as a distance preserver for any subset of the pairs and thus we have for any $v \in V$, $\mathsf{OPT}(S_v^{sink}) \leq \mathsf{OPT}$ and $\mathsf{OPT}(S_v^{source}) \leq \mathsf{OPT}$. Thus, using Theorem 1.9, the cost of the G' returned by Algorithm 4 is at most $|R| \cdot \tilde{O}(n^\varepsilon \cdot \mathsf{OPT}) \leq \tilde{O}(n^\varepsilon \cdot \beta \cdot \mathsf{OPT})$.

Using a hitting set argument very similar to Claim 2.1, we can see that with high probability, there is at least one sample v such that there is a $s \rightsquigarrow v \rightsquigarrow t$ path for every $(s,t) \in V \times V$ where $d_{G'}(s,v) + d_{G'}(v,t) = d_G(s,t)$.

The single-sink distance preserver gives us a $s \rightsquigarrow v$ path of length $d_G(s,v)$ and the single-source distance preserver gives us a $v \rightsquigarrow t$ path of length $d_G(v,t)$. Thus, thick pairs are resolved with high probability by the edges in G'. $\qquad\square$

3.2 Thin pairs

To resolve thin pairs, we start by redefining anti-spanners by ignoring the path costs in Definition 2.16.

Definition 3.2. A set $A \subseteq E$ is an anti-spanner for a demand pair $(s,t) \in E$ if $(V, E \setminus A)$ contains no feasible path from s to t. If no proper subset of anti-spanner A for (s,t) is an anti-spanner for (s,t), then A is minimal. The set of all minimal anti-spanners for all thin edges is denoted by $\mathcal{A}$.

Consider the following LP which is a slightly modified version of a similar LP from [10] (Fig. 1).

$$
\begin{aligned}
\min \quad & \sum_{e \in E} c(e) \cdot x_e \\
\text{subject to} \quad & \sum_{e \in A} x_e \geq 1 \quad \forall A \in \mathcal{A}, \\
& x_e \geq 0 \quad\quad\ \forall e \in E.
\end{aligned}
\tag{7}
$$

Let OPT denote the optimal solution to the LP. We can obtain this in a way identical to [10] as we only change the objective (which does not affect the separation oracle). Now, if $\{\hat{x}_e\}$ denotes the vector of x_e's in the solution to LP (7), then add every edge $e \in E$ to G' with probability $\min(\sqrt{n} \cdot \ln n \cdot \hat{x}_e, 1)$

We now state the following claim from [10] (Corollary 2.7).

Claim 3.3. *Given a feasible solution to LP (7), the rounding procedure produces a set of edges E'' that settles all thin pairs with high probability and has size at most $2\mathsf{OPT} \cdot \sqrt{n} \cdot \ln n$.*

Theorem 1.4. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized algorithm for* ALL-PAIR WEIGHTED DISTANCE PRESERVER *with approximation ratio $\tilde{O}(n^{1/2+\varepsilon})$, which succeeds in resolving all pairs in $V \times V$ with high probability.*

Proof of Theorem 1.4. Using Lemma 3.1 we can resolve all thick pairs with high probability with cost at most $\tilde{O}(n^{1/2+\varepsilon} \cdot \mathsf{OPT})$ by running Algorithm 4. Then, using Claim 3.3, we can solve and round LP (7) to resolve the thin pairs with high probability and cost $\tilde{O}(\mathsf{OPT} \cdot \sqrt{n})$. $\qquad\square$

Remark 3.4. Note that using Algorithm 4 for PAIRWISE WEIGHTED SPANNER (even for all-pair spanners) would not work because the cost of the single-source (sink) spanners that we need to add cannot be compared to the overall optimal solution. This is because, such single-source (sink) spanners cannot have a relaxation on the distances involved, unlike the optimal solution.

4 Online Weighted Spanners

In the online problem, the directed graph, the edge lengths, and the edge costs are given offline. The vertex pairs and the corresponding target distances arrive one at a time, in an online fashion, at each time stamp. The algorithm must irrevocably select edges at each time stamp and the goal is to minimize the cost, subject to the target distance constraints. We recall the definition for ONLINE PAIRWISE WEIGHTED SPANNER. For notation convenience, the vertex pair (s_i, t_i) denotes the i-th pair that arrives online.

Definition 1.5. ONLINE PAIRWISE WEIGHTED SPANNER
   Instance: A directed graph $G = (V, E)$ with edge costs $c : E \rightarrow \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \rightarrow \{1, 2, 3, ..., \mathsf{poly}(n)\}$, and vertex pairs $D = \{(s_i, t_i) \in V \times V \mid i \in [k]\}$ (k is unknown) with their corresponding pairwise distance bounds $Dist(s_i, t_i) \in \mathbb{Q}_{\geq 0}$ (where $Dist(s_i, t_i) \geq d_G(s_i, t_i)$) arrive online one at a time.
   Objective: Upon the arrival of (s_i, t_i) with $Dist(s_i, t_i)$, construct a min-cost subgraph H of G such that $d_H(s_i, t_i) \leq Dist(s_i, t_i)$ by irrevocably adding edges from E.

   We also consider the single-source online problem.

Definition 1.8. ONLINE SINGLE-SOURCE WEIGHTED SPANNER
   Instance: A directed graph $G = (V, E)$ with edge costs $c : E \rightarrow \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \rightarrow \{1, 2, 3, ..., \mathsf{poly}(n)\}$, and vertex pairs $D = \{(s, t_i) \mid t_i \in V, i \in [k]\}$ (k is unknown) with their corresponding pairwise distance bounds $Dist(s, t_i) \in \mathbb{Q}_{\geq 0}$ (where $Dist(s, t_i) \geq d_G(s, t_i)$) arrive online one at a time.
   Objective: Upon the arrival of (s, t_i) with $Dist(s, t_i)$, construct a min-cost subgraph H of G such that $d_H(s, t_i) \leq Dist(s, t_i)$ by irrevocably selecting edges from E.

   This section is dedicated to proving the following theorems.

Theorem 1.6. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized online algorithm for* ONLINE PAIRWISE WEIGHTED SPANNER *with competitive ratio $\tilde{O}(k^{1/2+\varepsilon})$, which succeeds in resolving all pairs in D with high probability.*

Theorem 1.9. *For any constant $\varepsilon > 0$, there is a polynomial-time randomized online algorithm for* ONLINE SINGLE-SOURCE WEIGHTED SPANNER *with approximation ratio $\tilde{O}(k^{\varepsilon})$, which succeeds in resolving all pairs in D with high probability.*

   The proof outline for Theorems 1.6 and 1.9 is as follows.

1. We first show that there exists an α-approximate solution consisting of distance-preserving weighted junction trees (see Definition 2.6). Here, $\alpha = O(\sqrt{k})$ for PAIRWISE WEIGHTED SPANNER and $\alpha = 1$ for SINGLE-SOURCE WEIGHTED SPANNER.

2. We slightly modify the online algorithm from [32] to find an online solution consisting of distance-preserving weighted junction trees by losing a factor of $\tilde{O}(k^{\varepsilon})$.

   The main difference between the online approach and the offline approach in Section 2 is that we cannot greedily remove partial solutions to settle the terminal pairs in the online setting. Instead, we construct a distance-preserving weighted junction tree solution in an online fashion.

Definition 4.1. A *distance-preserving weighted junction tree solution* is a collection of distance-preserving weighted junction trees rooted at different vertices, that satisfies all the terminal distance constraints.

We construct a distance-preserving weighted junction tree solution online and compare the online objective with the optimal distance-preserving weighted junction tree solution with objective value OPT_{junc}. The following theorem is essentially from [32] for the case when the edges have unit costs and lengths. However, the slight yet important modifications that we need when edges have arbitrary positive costs and integral lengths in $\mathsf{poly}(n)$ are not covered in [32]. For completeness, we show the proof in Appendix B.1.

Theorem 4.2. *For any constant $\varepsilon > 0$, there exists a polynomial-time randomized online algorithm for* ONLINE PAIRWISE WEIGHTED SPANNER *that constructs a distance-preserving weighted junction tree solution online with a cost at most $\tilde{O}(k^\varepsilon)\mathsf{OPT}_{junc}$ with high probability.*

With this theorem, we are ready to prove Theorems 1.6 and 1.9.

Proof for Theorems 1.6 and 1.9. Let OPT denote the cost of the optimal solution and α denote the ratio between OPT_{junc} and OPT. It suffices to show that $\alpha = O(\sqrt{k})$ for PAIRWISE WEIGHTED SPANNER and $\alpha = 1$ for SINGLE-SOURCE WEIGHTED SPANNER because Theorem 4.2 implies the existence of an $\tilde{O}(\alpha k^\varepsilon)$-competitive online algorithm.

To show that $\alpha = 1$ for SINGLE-SOURCE WEIGHTED SPANNER, let H be an optimal solution. We observe that H itself is a distance-preserving weighted junction tree rooted at the source s that is connected to all the k sinks, so $\alpha = 1$.

To show that $\alpha = O(\sqrt{k})$ for PAIRWISE WEIGHTED SPANNER, we use a density argument via a greedy procedure which implies an $O(\sqrt{k})$-approximate distance-preserving weighted junction tree solution. We recall the density notion in Section 2.2. The density of a distance-preserving weighted junction tree is its cost divided by the number of terminal pairs that it connects within the required distances.

Intuitively, we are interested in finding low-density distance-preserving weighted junction trees. We show that there always exists a distance-preserving weighted junction tree with density at most a $\sqrt{k}$ factor of the optimal density. The proof of Lemma 4.3 closely follows the one for the directed Steiner network problem in [18] and pairwise spanners [32] by considering whether there is a *heavy* vertex that lies in $s_i \rightsquigarrow t_i$ paths for distinct i or there is a simple path with low density. The case analysis also holds when there is a distance constraint for each (s_i, t_i). We provide the proof in Appendix B.2 for the sake of completeness.

Lemma 4.3. *There exists a distance-preserving weighted junction tree J with density at most $\mathsf{OPT}/\sqrt{k}$.*

Consider the procedure that finds a minimum density distance-preserving weighted junction tree in each iteration, and continues on the remaining disconnected terminal pairs. Suppose there are t iterations, and after iteration $j \in [t]$, there are n_j disconnected terminal pairs. Let $n_0 = k$ and $n_t = 0$. After each iteration, the minimum cost for connecting the remaining terminal pairs in the remaining graph is at most OPT, so the total cost of this procedure is upper-bounded by

$$\sum_{j=1}^{t} \frac{(n_{j-1} - n_j)\mathsf{OPT}}{\sqrt{n_{j-1}}} \leq \sum_{i=1}^{k} \frac{\mathsf{OPT}}{\sqrt{i}} \leq \int_{1}^{k+1} \frac{\mathsf{OPT}}{\sqrt{x}} dx = 2\mathsf{OPT}(\sqrt{k+1} - 1) = O(\sqrt{k})\mathsf{OPT}$$

where the first inequality uses the upper bound by considering the worst case when only one terminal pair is removed in each iteration of the procedure. □

Remark 4.4. The online algorithms imply efficient algorithms for the corresponding offline problems with the same approximation ratios. For offline algorithms, one can also use a greedy approach by iteratively removing the connected terminal pairs and extracting low-density distance-preserving weighted junction trees, with a poly-logarithmic factor improvement. The poly-logarithmic factor naturally appears from the challenge that the terminal pairs arrive online. This greedy approach is not applicable in the online setting.

5 Conclusion

In this paper, we presented algorithms for a variant of directed spanners that could also handle costs on edges, in addition to the more standard setting of edge lengths. The proof strategy for Theorem 1.2 follows a high-level structure that is similar to other results for directed Steiner forests, but involves significant obstacles in each part of the proof due to the addition of distance constraints. We overcome these obstacles by using the proper approaches. For example, the RESOURCE-CONSTRAINED SHORTEST PATH problem from [38] is carefully adapted for our specifics. We also needed to carefully adapt many other parts of the proof, such as the analysis of our junction-tree approximation and our rounding algorithm for the LPs, to fit the addition of distance constraints.

We also present online algorithms for ONLINE PAIRWISE WEIGHTED SPANNER and ONLINE SINGLE-SOURCE WEIGHTED SPANNER. We use our result for ONLINE SINGLE-SOURCE WEIGHTED SPANNER to solve a special case of PAIRWISE WEIGHTED SPANNER, namely, ALL-PAIR WEIGHTED DISTANCE PRESERVER, and obtain a significantly better approximation for that case.

We propose the following directions for future work:

- Is it possible to get a better analysis for the rounding algorithm for Theorem 1.2 as in [10]? This should improve the overall approximation factor for PAIRWISE WEIGHTED SPANNER in Theorem 1.2.

- Is there a hardness bound for PAIRWISE WEIGHTED SPANNER that is greater than the existing hardness bounds for Steiner forests and unit-cost spanners?

- Is there a better approximation factor for all-pair weighted spanners, i.e., an instance of PAIRWISE WEIGHTED SPANNER where $D = V \times V$?

- Can we get a result for pairwise weighted distance preserver that is better than using the PAIRWISE WEIGHTED SPANNER results in Theorems 1.2?

References

[1] ABBOUD, A., AND BODWIN, G. Reachability preservers: New extremal bounds and approximation algorithms. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)* (2018), SIAM, pp. 1865–1883.

[2] AHMED, R., BODWIN, G., SAHNEH, F. D., HAMM, K., JEBELLI, M. J. L., KOBOUROV, S., AND SPENCE, R. Graph spanners: A tutorial review. *Computer Science Review 37* (2020), 100253.

[3] ALON, N., AWERBUCH, B., AZAR, Y., BUCHBINDER, N., AND NAOR, J. A general approach to online network optimization problems. *ACM Transactions on Algorithms (TALG) 2*, 4 (2006), 640–660.

[4] ALON, N., AND SCHIEBER, B. Optimal preprocessing for answering on-line product queries. Tech. rep., 1987.

[5] ANTONAKOPOULOS, S. Approximating directed buy-at-bulk network design. In *International Workshop on Approximation and Online Algorithms* (2010), Springer, pp. 13–24.

[6] AWASTHI, P., JHA, M., MOLINARO, M., AND RASKHODNIKOVA, S. Testing lipschitz functions on hypergrid domains. *Algorithmica 74*, 3 (2016), 1055–1081.

[7] AWERBUCH, B. Communication-time trade-offs in network synchronization. In *Proceedings of the 4th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)* (New York, NY, USA, 1985), p. 272–276.

[8] BASWANA, S., AND KAVITHA, T. Faster algorithms for all-pairs approximate shortest paths in undirected graphs. *SIAM Journal on Computing 39*, 7 (2010), 2865–2896.

[9] BATENI, M., AND HAJIAGHAYI, M. Euclidean prize-collecting steiner forest. *Algorithmica 62*, 3-4 (2012), 906–929.

[10] BERMAN, P., BHATTACHARYYA, A., MAKARYCHEV, K., RASKHODNIKOVA, S., AND YAROSLAVT-SEV, G. Approximation algorithms for spanner problems and directed steiner forest. *Information and Computation 222* (2013), 93–107.

[11] BHATTACHARYYA, A., GRIGORESCU, E., JUNG, K., RASKHODNIKOVA, S., AND WOODRUFF, D. P. Transitive-closure spanners. *SIAM Journal on Computing 41*, 6 (2012), 1380–1425.

[12] BODWIN, G. New results on linear size distance preservers. *SIAM Journal on Computing 50*, 2 (2021), 662–673.

[13] BORRADAILE, G., KLEIN, P. N., AND MATHIEU, C. A polynomial-time approximation scheme for euclidean steiner forest. *ACM Transactions on Algorithms (TALG) 11*, 3 (2015), 1–20.

[14] CHAKRABARTY, D., ENE, A., KRISHNASWAMY, R., AND PANIGRAHI, D. Online buy-at-bulk network design. *SIAM Journal on Computing 47*, 4 (2018), 1505–1528.

[15] CHARIKAR, M., CHEKURI, C., CHEUNG, T.-Y., DAI, Z., GOEL, A., GUHA, S., AND LI, M. Approximation algorithms for directed steiner problems. *Journal of Algorithms 33*, 1 (1999), 73–91.

[16] CHAWLA, S., ROUGHGARDEN, T., AND SUNDARARAJAN, M. Optimal cost-sharing mechanisms for steiner forest problems. In *International Workshop on Internet and Network Economics* (2006), Springer, pp. 112–123.

[17] CHECHIK, S. Approximate distance oracles with improved bounds. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)* (2015), R. A. Servedio and R. Rubinfeld, Eds., ACM, pp. 1–10.

[18] CHEKURI, C., EVEN, G., GUPTA, A., AND SEGEV, D. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Transactions on Algorithms (TALG) 7*, 2 (2011), 1–17.

[19] CHLAMTÁČ, E., DINITZ, M., KORTSARZ, G., AND LAEKHANUKIT, B. Approximating spanners and directed steiner forest: Upper and lower bounds. *ACM Transactions on Algorithms (TALG) 16*, 3 (2020), 1–31.

[20] COWEN, L., AND WAGNER, C. G. Compact roundtrip routing in directed networks. *Journal on Algorithms 50*, 1 (2004), 79–95.

[21] DINITZ, M., AND KRAUTHGAMER, R. Directed spanners via flow-based linear programs. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)* (2011), pp. 323–332.

[22] DINITZ, M., AND KRAUTHGAMER, R. Fault-tolerant spanners: better and simpler. In *Proceedings of the 30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)* (2011), pp. 169–178.

[23] DINITZ, M., AND ZHANG, Z. Approximating low-stretch spanners. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete algorithms (SODA)* (2016), SIAM, pp. 821–840.

[24] DOR, D., HALPERIN, S., AND ZWICK, U. All-pairs almost shortest paths. *SIAM Journal on Computing 29*, 5 (2000), 1740–1759.

[25] ELKIN, M. Computing almost shortest paths. *ACM Transactions on Algorithms (TALG) 1*, 2 (2005), 283–323.

[26] ELKIN, M., AND PELEG, D. The client-server 2-spanner problem with applications to network design. In *SIROCCO 8, Proceedings of the 8th International Colloquium on Structural Information and Communication Complexity, Vall de Núria, Girona-Barcelona, Catalonia, Spain, 27-29 June, 2001* (2001), F. Comellas, J. Fàbrega, and P. Fraigniaud, Eds., vol. 8 of *Proceedings in Informatics*, Carleton Scientific, pp. 117–132.

[27] ELKIN, M., AND PELEG, D. The hardness of approximating spanner problems. *Theory of Computing Systems 41*, 4 (2007), 691–729.

[28] FELDMAN, M., KORTSARZ, G., AND NUTOV, Z. Improved approximation algorithms for directed steiner forest. *Journal of Computer and System Sciences 78*, 1 (2012), 279–292.

[29] FILTSER, A. Hop-constrained metric embeddings and their applications. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)* (2021), IEEE, pp. 492–503.

[30] FLEISCHER, L., KÖNEMANN, J., LEONARDI, S., AND SCHÄFER, G. Simple cost sharing schemes for multicommodity rent-or-buy and stochastic steiner tree. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)* (2006), pp. 663–670.

[31] FOMIN, F. V., GOLOVACH, P. A., LOCHET, W., MISRA, P., SAURABH, S., AND SHARMA, R. Parameterized complexity of directed spanner problems. *Algorithmica 84*, 8 (2022), 2292–2308.

[32] GRIGORESCU, E., LIN, Y.-S., AND QUANRUD, K. Online directed spanners and steiner forests. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)* (2021), Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[33] GUPTA, A., KUMAR, A., PÁL, M., AND ROUGHGARDEN, T. Approximation via cost-sharing: a simple approximation algorithm for the multicommodity rent-or-buy problem. In *44th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2003. Proceedings.* (2003), IEEE, pp. 606–615.

[34] HAEUPLER, B., HERSHKOWITZ, D. E., AND ZUZIC, G. Tree embeddings for hop-constrained network design. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing* (2021), pp. 356–369.

[35] HAJIAGHAYI, M. T., KORTSARZ, G., AND SALAVATIPOUR, M. R. Approximating buy-at-bulk and shallow-light k-steiner trees. *Algorithmica 53* (2009), 89–103.

[36] HASSIN, R. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations research 17*, 1 (1992), 36–42.

[37] HELVIG, C. S., ROBINS, G., AND ZELIKOVSKY, A. An improved approximation scheme for the group steiner problem. *Networks: An International Journal 37*, 1 (2001), 8–20.

[38] HORVÁTH, M., AND KIS, T. Multi-criteria approximation schemes for the resource constrained shortest path problem. *Optimization Letters 12*, 3 (2018), 475–483.

[39] KHULLER, S., RAGHAVACHARI, B., AND YOUNG, N. Approximating the minimum equivalent digraph. *SIAM Journal on Computing 24*, 4 (1995), 859–872.

[40] KHURANA, V., PENG, J., CHUNG, C. Y., AULUCK, P. K., FANNING, S., TARDIFF, D. F., BARTELS, T., KOEVA, M., EICHHORN, S. W., BENYAMINI, H., ET AL. Genome-scale networks link neurodegenerative disease genes to α-synuclein through specific molecular pathways. *Cell systems 4*, 2 (2017), 157–170.

[41] KOGAN, S., AND PARTER, M. Having hope in hops: New spanners, preservers and lower bounds for hopsets. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* (2022), IEEE, pp. 766–777.

[42] KÖNEMANN, J., LEONARDI, S., SCHÄFER, G., AND VAN ZWAM, S. From primal-dual to cost shares and back: a stronger lp relaxation for the steiner forest problem. In *International Colloquium on Automata, Languages, and Programming* (2005), Springer, pp. 930–942.

[43] KÖNEMANN, J., LEONARDI, S., SCHÄFER, G., AND VAN ZWAM, S. H. A group-strategyproof cost sharing mechanism for the steiner forest game. *SIAM Journal on Computing 37*, 5 (2008), 1319–1341.

[44] KORTSARZ, G. On the hardness of approximating spanners. *Algorithmica 30* (2001), 432–450.

[45] LORENZ, D. H., AND RAZ, D. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters 28*, 5 (2001), 213–219.

[46] MARATHE, M. V., SUNDARAM, R., RAVI, S., ROSENKRANTZ, D. J., AND HUNT III, H. B. Bicriteria network design problems. *Journal of algorithms 28*, 1 (1998), 142–171.

[47] PACHOCKI, J., RODITTY, L., SIDFORD, A., TOV, R., AND WILLIAMS, V. V. Approximating cycles in directed graphs: Fast algorithms for girth and roundtrip spanners. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete algorithms (SODA)* (2018), A. Czumaj, Ed., SIAM, pp. 1374–1392.

[48] PATRASCU, M., AND RODITTY, L. Distance oracles beyond the Thorup-Zwick bound. *SIAM Journal on Computing 43*, 1 (2014), 300–311.

[49] PELEG, D., AND SCHÄFFER, A. A. Graph spanners. *Journal of Graph Theory 13*, 1 (1989), 99–116.

[50] PELEG, D., AND ULLMAN, J. D. An optimal synchronizer for the hypercube. *SIAM Journal on Computing 18*, 4 (1989), 740–747.

[51] PIRHAJI, L., MILANI, P., LEIDL, M., CURRAN, T., AVILA-PACHECO, J., CLISH, C. B., WHITE, F. M., SAGHATELIAN, A., AND FRAENKEL, E. Revealing disease-associated pathways by network integration of untargeted metabolomics. *Nature methods 13*, 9 (2016), 770–776.

[52] RODITTY, L., THORUP, M., AND ZWICK, U. Roundtrip spanners and roundtrip routing in directed graphs. *ACM Transactions on Algorithms (TALG) 4*, 3 (2008), 29:1–29:17.

[53] ROUGHGARDEN, T., AND SUNDARARAJAN, M. Optimal efficiency guarantees for network design mechanisms. In *International Conference on Integer Programming and Combinatorial Optimization* (2007), Springer, pp. 469–483.

[54] VETTA, A. Approximating the minimum strongly connected subgraph via a matching lower bound. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete algorithms (SODA)* (2001), pp. 417–426.

[55] YAO, A. C.-C. Space-time tradeoff for answering range queries (extended abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing (STOC)* (1982).

[56] ZELIKOVSKY, A. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica 18*, 1 (1997), 99–110.

A Missing Proofs in Section 2

A.1 Proof of Claim 2.1

We recall Claim 2.1 and present its proof here.

Claim 2.1. *Algorithm 1 selects a set of samples R such that with high probability any given thick pair (s, t) has at least one vertex from its local graph in R.*

Proof. Because (s, t) is a thick pair, we have at least n/β vertices in its local graph by definition. Let $\text{EVENT}(s, t)$ be the event that no vertex in the local graph of (s, t) is sampled. We have that

$$\mathbf{Pr}[\text{EVENT}(s, t)] = \left(1 - \frac{n/\beta}{n}\right)^{3\beta \ln n} = \left(1 - \frac{1}{\beta}\right)^{3\beta \ln n} \leq \exp\left(-\frac{3\beta \ln n}{\beta}\right) = \frac{1}{n^3}.$$

Taking the union bound over all demand pairs in D where $|D| \leq n^2$, the probability that there is one $(s, t) \in D$ whose local graph does not include a sampled vertex is at most $1/n$. The probability that any $(s, t) \in D$ has at least one vertex sampled is at least $1 - 1/n$. $\square$

A.2 Weighted junction tree with distance constraints

We recall Lemma 2.9 and present its proof here. But we first prove a result for the same problem of the junction trees in Lemma 2.9 where we assume all edges have unit lengths (but costs are general) and then reduce the polynomial integral length version to that problem.

Lemma A.1. *For any constant $\delta > 0$, there is a polynomial-time approximation algorithm for the minimum density distance-preserving weighted junction tree as long as the edge lengths are uniform. In other words, there is a polynomial-time algorithm which, given a weighted directed n vertex graph $G = (V, E)$ where each edge $e \in E$ has a cost $c(e) \in \mathbb{R}_{\geq 0}$ and unit length $\ell(e) = 1$, terminal pairs $P \subseteq V \times V$, and distance bounds $Dist : P \to \mathbb{N}$ (where $Dist(s, t) \geq d_G(s, t)$) for every terminal pair $(s, t) \in P$, approximates the following problem to within an $O(n^\delta)$ factor:*

- *Find a non-empty set of edges $F \subseteq E$ minimizing the ratio:*

$$\min_{r \in V} \frac{\sum_{e \in F} c(e)}{|\{(s, t) \in P | d_{F,r}(s, t) \leq Dist(s, t)\}|} \tag{8}$$

where $d_{F,r}(s, t)$ is the length of the shortest path using edges in F which connects s to t while going through r (if such a path exists).[6]

Proof. In this proof, we follow the structure for the proof of Theorem 5.1 in [19] very closely. We first define the MINIMUM DENSITY LABEL COVER problem, then we reduce an instance of our problem to an instance of the MINIMUM DENSITY LABEL COVER problem using a graph construction. We also state the height reduction lemma from [18, 19] and use it on our instance of MINIMUM DENSITY LABEL COVER. Finally, we use Lemma (5.4) from [19] to prove our result. We do this procedure with every possible root $r \in V$ and take the distance-preserving weighted junction tree with the minimum density distance-preserving weighted junction tree among all possible choices of r.

We start by defining the MINIMUM DENSITY LABEL COVER problem to which we will reduce our problem.

Definition A.2. In the MINIMUM DENSITY LABEL COVER problem, we are given a directed graph $G = (V, E)$, nonnegative edge costs $w : E \to \mathbb{R}_{\geq 0}$, a collection of set pairs $B \subseteq 2^V \times 2^V$, and for each pair $(S, T) \in B$, a relation $R(S, T) \subseteq S \times T$. The goal is to find a set of edges $F \subseteq E$ minimizing the ratio

$$\frac{\sum_{e \in F} w(e)}{|\{(S, T) \in B \mid \exists (s, t) \in R(S, T) : \ F \text{ contains an } s \rightsquigarrow t \text{ path }\}|}.$$

We also need the height reduction lemma from [18, 19].

Lemma A.3. *(Height Reduction) Let $G = (V, E)$ be an edge-weighted directed graph with edge weights $w : E \to \mathbb{R}_{\geq 0}$, let $r \in V$ be a source vertex of G, and let $\sigma > 0$ be some parameter. Then we can efficiently construct an edge-weighted undirected tree $\hat{T}_r$ rooted at $\hat{r}$ of height σ and size $|V|^{O(\sigma)}$ together with edge weights $\hat{w} : E(\hat{T}_r) \to \mathbb{R}_{\geq 0}$, and a vertex mapping $\psi : V(\hat{T}_r) \to V(G)$, such that*

---

[6]In [19], the numerator is $|F|$. The difference occurs because they effectively have $c(e) = 1$ for all $e \in E$.

- *For any arboresence $J \subseteq G$ rooted at r, and terminal set $S \subseteq J$, there exists a tree $\hat{J} \subseteq \hat{T}_r$ rooted at $\hat{r}$ such that letting $L(J)$ and $L(\hat{J})$ be the set of leaves of J and $\hat{J}$, respectively, we have $L(J) = \psi(L(\hat{J}))$. Moreover, $w(J) \leq O(\sigma |L|^{1/\sigma} \cdot \hat{w}(\hat{J}))$.*

- *Given any tree $\hat{J} \subseteq \hat{T}_r$ rooted at $\hat{r}$, we can efficiently find an arboresence $J \subseteq G$ rooted at r such that, for leaf sets $L(J)$ and $L(\hat{J})$ as above, we have $L(J) = \psi(L(\hat{J}))$, and moreover, $w(J) \leq \hat{w}(\hat{J})$.*

We do the following procedure using every possible root $r \in V$ and then take the minimum density distance-preserving weighted junction tree.

For a specific r, we turn our distance problem into a connectivity problem using the following reduction which closely matches [19] except for one change. We construct a layered directed graph with vertices:

$$V_r = ((V \setminus r) \times \{-n+1, \ldots, -2, -1, 1, 2, \ldots, n-1\}) \cup \{(r, 0)\}$$

and

$$E_r = \{((u, i)(v, i+1)) | (u, i), (v, i+1) \in V_r, (u, v) \in E\}.$$

We set the edge weights for $e \in E_r$ connecting some $((u, i)(v, i+1))$ to $w(e) = c((u, v))$. This is the only change we make from [19] and it has very little impact on the overall proof.

For every terminal pair $(s, t) \in P$ with distance bound $Dist(s, t)$, add new vertices $(s^t, -i)$ and (t^s, j) for all $i, j \geq 0$ such that $(s, -i), (t, j) \in V_r$, and for all such i and j add zero-weight edges $((s^t, -i)(s, -i))$ and $((t, j)(t^s, j))$. Denote this graph as G_r, now for every terminal pair $(s, t) \in P$ define terminal sets $S_{s,t} = \{(s^t, -i) | i \geq 0\} \cap V(G_r)$ and $T_{s,t} = \{(t^s, j) | j \geq 0\} \cap V(G_r)$ and relation $R_{s,t} = \{(s^t, -i), (t^s, j) \in S_{s,t} \times T_{s,t} | i + j \leq Dist(s, t)\}$.

In this construction, for every terminal pair $(s, t) \in P$ and label i, there is a bijection between paths of length i from s to r in G, and paths from $(s^t, -i)$ to $(r, 0)$ in G_r, and similarly a bijection between paths of length i from r to t in G, and paths from $(r, 0)$ to (t^s, i) in G_r. Now, to keep track of path lengths in G we can just connect the appropriate terminal pairs in G_r. This construction also creates disjoint terminal pairs, by creating a separate copy s^t of s for every terminal pair $(s, t) \in P$ that s participates in (and similarly for terminals t).
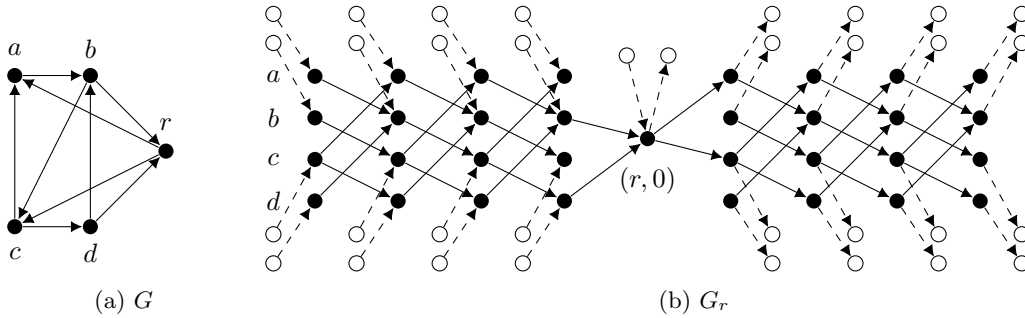

(a) G (b) G_r

Figure 1: Construction of G_r given $G = (V, E)$ and $r \in V$. The dotted lines in G_r have zero weight while the solid lines in G_r have the same weight as the corresponding edges in G. The figure is from [32].

We finally generate a new graph G' which is made up of two graphs G_+ and G_- which are copies of G_r intersecting only in the vertex r. For every node $u \in V$, we use u_+, u_- to denote the copies of u in G_+, G_- respectively.

The following lemma (which is also in [19]) follows from our construction.

Lemma A.4. *For any $f > 0$, and set of terminal pairs $P' \subseteq P$, there exists an edge set $F \subseteq E(G')$ of weight $\sum_{e \in F} w(e) \leq f$ containing a path of length at most $Dist(s, t)$ from s_- to t_+ for every $(s, t) \in P'$ iff there exists a junction tree $J \subseteq E(G_r)$ of weight $w(J) \leq f$ such that for every terminal pair $(s, t) \in P'$, J contains leaves $(s^t, -i), (t^s, j)$ such that $((s^t, -i)(t^s, j)) \in R_{s,t}$. Morever, given such a junction tree J, we can efficiently find a corresponding edge set F.*

Now, to prove Lemma A.1, we just need to show that we can achieve an $O(n^\varepsilon)$ approximation for the MINIMUM DENSITY LABEL COVER instance $(G_r, w, \{(S_{s,t}, T_{s,t}), R_{s,t} \mid (s,t) \in P\})$ obtained from our reduction. We apply Lemma A.3 to our weighted graph (G_r, w) with constant parameter $\sigma > 1/\varepsilon$, and obtain a shallow tree $\hat{T}_r$ with weights $\hat{w}$, and a mapping $\psi : V(\hat{T}_r) \to V(G_r)$. All terminals $(s^t, -i), (t^s, j)$ in G_r are now represented by sets of terminals $\psi^{-1}((s^t, -i))$ and $\psi^{-1}((t^s, j))$, respectively in $V(\hat{T}_r)$. We can extend the relation $R_{s,t}$ in the natural way to

$$\hat{R}_{s,t} = \{(\hat{s}, \hat{t}) \in \psi^{-1}(S_{s,t}) \times \psi^{-1}(T_{s,t}) \mid (\psi(\hat{s}), \psi(\hat{t})) \in R_{s,t}\}.$$

Thus, using Lemmas A.3 and A.4, it suffices to show the following lemma which is already proved in [19].

Lemma A.5. *There exists a polynomial time algorithm which, in the above setting, gives an $O(\log^3 n)$ approximation for the following problem:*

- *Find a tree $T \subseteq \hat{T}_r$ minimizing the ratio*

$$\frac{\hat{w}(E(T))}{|\{(s,t) \in P \mid \exists (\hat{s}, \hat{t}) \in \hat{R}(S,T) : \ T \text{ contains an } \hat{s}\text{-}\hat{t} \text{ path }\}|}.$$

$\square$

We now adapt the above lemma to have general lengths with a very simple reduction.

Lemma 2.9. *For any constant $\varepsilon > 0$, there is a polynomial-time approximation algorithm for the minimum density distance-preserving junction tree as long as the edge lengths are integral and polynomial in n. In other words, there is a polynomial time algorithm which, given a weighted directed n vertex graph $G = (V, E)$ where each edge $e \in E$ has a cost $c(e) \in \mathbb{R}_{\geq 0}$ and integral length $\ell(e) \in \{1, 2, ..., \mathsf{poly}(n)\}$, terminal pairs $P \subseteq V \times V$, and distance bounds $Dist : P \to \mathbb{N}$ (where $Dist(s,t) \geq d_G(s,t)$) for every terminal pair $(s,t) \in P$, approximates the following problem to within an $O(n^\varepsilon)$ factor:*

- *Find a non-empty set of edges $F \subseteq E$ minimizing the ratio:*

$$\min_{r \in V} \frac{\sum_{e \in F} c(e)}{|\{(s,t) \in P \mid d_{F,r}(s,t) \leq Dist(s,t)\}|} \tag{1}$$

where $d_{F,r}(s,t)$ is the length of the shortest path using edges in F which connects s to t while going through r (if such a path exists).

Proof. Let the edge lengths in our instance be in $O(n^\lambda)$ for some constant $\lambda > 0$. This assumption is valid because the edge lengths are in $\{1, 2, ..., \mathsf{poly}(n)\}$. Given any integer and polynomial length instance, we can reduce it to an instance with unit lengths as follows. For any edge $e = (u, v) \in E$, with length $\ell(e)$, break it into $\ell(e)$ new edges which are connected as a path that starts at u and ends at v. Each of these new edges will have cost $c(e)/\ell(e)$ and be of unit length. We observe that solving this new instance of will give us a solution for our original problem. The whole reduction can be done in polynomial time and size because the newly created graph will have $O(n^\lambda \cdot |V|)$ vertices and $O(n^\lambda \cdot |E|)$ edges. Thus by setting $\delta = \varepsilon/(\lambda + 1)$ in Lemma A.1, we prove this lemma. $\square$

B   Missing Proofs in Section 4

B.1   Missing proof for Theorem 4.2

Theorem 4.2. *For any constant $\varepsilon > 0$, there exists a polynomial-time randomized online algorithm for* ONLINE PAIRWISE WEIGHTED SPANNER *that constructs a distance-preserving weighted junction tree solution online with a cost at most $\tilde{O}(k^\varepsilon)\mathsf{OPT}_{junc}$ with high probability.*

26

Proof. It suffices to show that the theorem holds even for ONLINE PAIRWISE WEIGHTED SPANNER when all the edges have unit lengths. Suppose we have a black-box online algorithm when edges have unit lengths. We can construct an online algorithm for ONLINE PAIRWISE WEIGHTED SPANNER by using the black box as follows. We decompose each edge $e \in E$ into a path consisting of $\ell(e)$ edges, each with unit length and cost $c(e)/\ell(e)$. The terminal vertices correspond to the endpoint vertices of the paths for the decomposed edges. The number of vertices in this graph is still polynomial in n because the edge lengths are polynomial in n. A black-box online algorithm on graphs with unit length edges therefore implies Theorem 4.2.

The proof for Theorem 4.2 on graphs with unit edge lengths is a slight modification from the $\tilde{O}(k^{1/2+\varepsilon})$-competitive online algorithm on graphs with unit edge lengths and costs in [32]. The outline is as follows.

1. We consider ONLINE PAIRWISE WEIGHTED SPANNER on a graph G' consisting of disjoint layered graphs constructed from the original graph G by losing a constant factor. The layers allow us to capture the distance constraints. This allows us to further reduce the problem to ONLINE STEINER LABEL COVER on G'.

2. We further consider ONLINE STEINER LABEL COVER on an undirected forest H constructed from G' with a loss of an $O(k^\varepsilon)$ factor via the height reduction technique introduced in [18, 37].

We start with the definition for STEINER LABEL COVER introduced in [32].

Definition B.1. In the *Steiner label cover* problem, we are given a (directed or undirected) graph $G = (V, E)$, non-negative edge costs $w : E \to \mathbb{R}_{\geq 0}$, and a collection of k disjoint vertex subset pairs (S_i, T_i) for $i \in [k]$ where $S_i, T_i \subseteq V$ and $S_i \cap T_i = \emptyset$. Each pair is associated with a relation (set of permissible pairs) $R_i \subseteq S_i \times T_i$. The goal is to find a subgraph $F = (V, E')$ of G, such that 1) for each $i \in [k]$, there exists $(s, t) \in R_i$ such that there is an $s \rightsquigarrow t$ path in F, and 2) the cost $\sum_{e \in E'} w(e)$ is minimized.

In ONLINE STEINER LABEL COVER, (S_i, T_i) and R_i arrive online, and the goal is to irrevocably select edges to satisfy the first requirement and also approximately minimize the cost.

Given an instance of ONLINE PAIRWISE WEIGHTED SPANNER on a graph with unit edge lengths, we construct the graph G' by following the construction of G_r in the proof of Lemma 2.9 in Appendix A.2 and taking the union of G_r for all $r \in V$. That is, we *simultaneously* consider all the vertices $r \in V$ as a candidate root for the distance-preserving weighted junction tree solution while constructing the online solution.

Recall that $G_r = (V_r, E_r)$ is constructed as follows (see Figure 1 for an example):

$$V_r = ((V \setminus r) \times \{-n+1, \ldots, -2, -1, 1, 2, \ldots, n-1\}) \cup \{(r, 0)\}$$

and

$$E_r = \{((u,i), (v, i+1)) \mid (u,i), (v, i+1) \in V_r, (u,v) \in E\}.$$

For each edge $e = ((u,i), (v, i+1)) \in E_r$, let the edge weight be $w(e) = c((u,v))$. This is the only change we make from [32] and it has very little impact on the overall proof. For each vertex $(u, -j)$ (respectively (u, j)) in V_r where $j \in [n-1] \cup \{0\}$, add a vertex $(u^-, -j)$ (respectively (u^+, j)), and create an edge $(u^-, -j) \to (u, -j)$ (respectively $(u, j) \to (u^+, j)$) with weight 0. This concludes the construction of G_r.

Let G' be the disjoint union of G_r for $r \in V$. Given an ONLINE PAIRWISE WEIGHTED SPANNER instance on G, we can construct an ONLINE STEINER LABEL COVER instance on G'. For explicitness, we denote the vertex (u, j) in V_r by $(u, j)_r$. For each (s_i, t_i), let $S_i = \{(s_i^-, -j)_r \mid j > 0, r \in V \setminus \{s_i\}\} \cup (s_i^-, 0)_{s_i}$, $T_i = \{(t_i^+, j)_r \mid j > 0, r \in V \setminus \{t_i\}\} \cup (t_i^+, 0)_{t_i}$, and $R_i = \{((s_i^-, -j_s)_r, (t_i^+, j_t)_r) \mid r \in V, j_s + j_t \leq Dist(s_i, t_i)\}$. Intuitively, a copy of s_i and a copy of t_i belongs to the relation R_i if 1) they are connected by a distance-preserving weighted junction tree with the same root r, and 2) the distance between them is at most $Dist(s_i, t_i)$ in this distance-preserving weighted junction tree.

We can recover a solution for ONLINE PAIRWISE WEIGHTED SPANNER on G with at most the cost of the corresponding ONLINE STEINER LABEL COVER solution in G'. Suppose we have the ONLINE STEINER LABEL COVER solution in G', for each selected edge $(u, j) \to (v, j+1)$ in G_r, we select $u \to v$ for the pairwise spanner solution in G. The optimum of the Steiner label cover solution in G' is at most $2\mathsf{OPT}_{junc}$. In each distance-preserving weighted, each edge is used at most once in the in-arborescence and at most once in the out-arborescence. The following claim is from [32].

Claim B.2. *The optimal value of the* STEINER LABEL COVER *problem in G' is at most* $2\mathit{OPT}_{junc}$.

The remaining is to show how height reduction is used to find a solution for ONLINE STEINER LABEL COVER on G'. We recall Lemma A.3.

Lemma A.3. *(Height Reduction) Let $G = (V, E)$ be an edge-weighted directed graph with edge weights $w : E \to \mathbb{R}_{\geq 0}$, let $r \in V$ be a source vertex of G, and let $\sigma > 0$ be some parameter. Then we can efficiently construct an edge-weighted undirected tree $\hat{T}_r$ rooted at $\hat{r}$ of height σ and size $|V|^{O(\sigma)}$ together with edge weights $\hat{w} : E(\hat{T}_r) \to \mathbb{R}_{\geq 0}$, and a vertex mapping $\psi : V(\hat{T}_r) \to V(G)$, such that*

- *For any arboresence $J \subseteq G$ rooted at r, and terminal set $S \subseteq J$, there exists a tree $\hat{J} \subseteq \hat{T}_r$ rooted at $\hat{r}$ such that letting $L(J)$ and $L(\hat{J})$ be the set of leaves of J and $\hat{J}$, respectively, we have $L(J) = \psi(L(\hat{J}))$. Moreover, $w(J) \leq O(\sigma |L|^{1/\sigma} \cdot \hat{w}(\hat{J}))$.*

- *Given any tree $\hat{J} \subseteq \hat{T}_r$ rooted at $\hat{r}$, we can efficiently find an arboresence $J \subseteq G$ rooted at r such that, for leaf sets $L(J)$ and $L(\hat{J})$ as above, we have $L(J) = \psi(L(\hat{J}))$, and moreover, $w(J) \leq \hat{w}(\hat{J})$.*

We construct a tree H_r from G_r by the height reduction technique. We recall that G_r is a layered graph with a center vertex $(r, 0)$. Let G_r^- denote the subgraph of G_r induced by the vertex set

$$\left(\bigcup_{v \in V \setminus \{r\}} \{v, v^-\} \times \{-j \mid j \in [n-1]\} \right) \cup \{(r, 0), (r^-, 0)\},$$

and similarly, let G_r^+ denote the subgraph of G_r induced by the vertex set

$$\left(\bigcup_{v \in V \setminus \{r\}} \{v, v^+\} \times [n-1] \right) \cup \{(r, 0), (r^+, 0)\}.$$

By Lemma A.3, with $(r, 0)$ being the root, we can construct a tree T_r^- rooted at r^- for G_r^- which approximately preserves the cost of any in-arborescence rooted at $(r, 0)$ in G_r^- by a subtree in T_r^-, and similarly a tree T_r^+ rooted at r^+ for G_r^+ which approximately preserves the cost of any out-arborescence rooted at $(r, 0)$ in G_r^+ by a subtree in T_r^+. We further add a super root r' and edges $\{r', r^-\}$ and $\{r', r^+\}$ both with weight 0. This concludes the construction of the weighted tree H_r (see Figure 2 for an illustration).
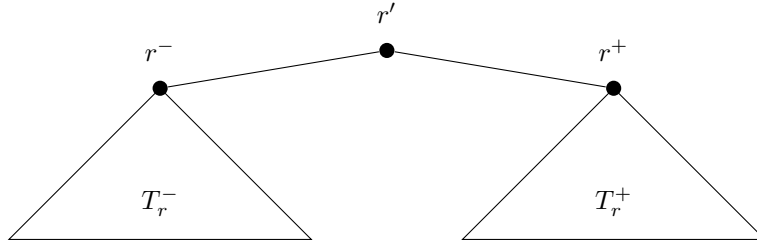

Figure 2: Construction of H_r. The figure is from [32].

Let H be the disjoint union of H_r for $r \in V$. We show that we can achieve an $\tilde{O}(k^\varepsilon)$-approximation for the Steiner label cover problem on graph H in an online manner. We set $\sigma = \lceil 1/\varepsilon \rceil$ and apply Lemma A.3 to obtain the weight $\hat{w} : E(H) \to \mathbb{R}_{\geq 0}$ and the mapping $\Psi_r : V(H_r) \to V(G_r)$ for all $r \in V$. Let Ψ be the union of the mappings for all $r \in V$. For each pair (s_i, t_i) in the original graph G, we recall that we focus on the vertex subset pair (S_i, T_i) of G' and its relation R_i which captures the distance requirement.

To establish the correspondence between the Steiner label cover instances, we clarify the mapping between the leaves of the arborescences in G' and H. Given a vertex $(s_i^-, -j)_r$ in $V(G')$ with a non-negative j, let $\Psi^{-1}((s_i^-, -j)_r)$ denote the set of leaves in T_r^- that maps to $(s_i^-, -j)_r$ by Ψ. Similarly, given a vertex $(t_i^+, j)_r$

in $V(G')$ with a non-negative j, let $\Psi^{-1}((t_i^+, j)_r)$ denote the set of leaves in T_r^+ that maps to $(t_i^+, j)_r$ by Ψ. The mapping Ψ^{-1} naturally defines the terminal sets of interest $\hat{S}_i := \Psi^{-1}(S_i) = \{\hat{s} \in V(H) \mid \Psi(\hat{s}) \in S_i\}$ and $\hat{T}_i := \Psi^{-1}(T_i) = \{\hat{t} \in V(H) \mid \Psi(\hat{t}) \in T_i\}$. The relation is also naturally defined: $\hat{R}_i := \{(\hat{s}, \hat{t}) \in \hat{S}_i \times \hat{T}_i \mid (\Psi(\hat{s}), \Psi(\hat{t})) \in R_i\}$. We note that for $\hat{s} \in \hat{S}_i$ and $\hat{t} \in \hat{T}_i$, $(\hat{s}, \hat{t})$ belongs to $\hat{R}_i$ only when $\hat{s}$ and $\hat{t}$ belong to the same tree H_r (see Figure 3 for an illustration).
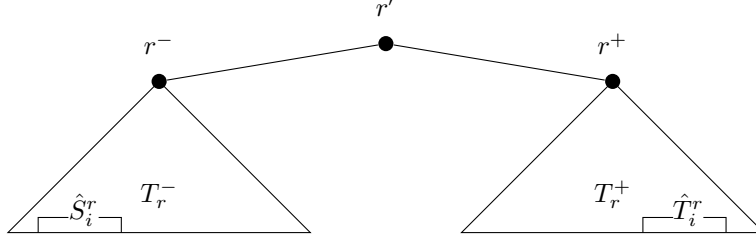

Figure 3: From ONLINE PAIRWISE WEIGHTED SPANNER to ONLINE STEINER LABEL COVER. In G, consider connecting s_i and t_i by using a distance-preserving weighted junction tree rooted at r. In H_r, we consider terminal sets $\hat{S}_i^r := \hat{S}_i \cap H_r$ and $\hat{T}_i^r := \hat{T}_i \cap H_r$. A distance-preserving weighted junction tree in G rooted at r corresponds to a subtree with leaves that belongs to $\hat{S}_i^r \cup \hat{T}_i^r$. The set $\hat{R}_i^r := \hat{R}_i \cap (\hat{S}_i^r \times \hat{T}_i^r)$ defines the admissible pairs, which maintains the distance requirement $Dist(s_i, t_i)$ in the layered graph G_r. The figure is from [32].

We use the following theorem from [32].

Theorem B.3. *For the* ONLINE STEINER LABEL COVER *problem on* $H, \hat{S}_i, \hat{T}_i, \hat{R}_i$, *there is a randomized polynomial-time algorithm with competitive ratio* $\mathsf{polylog}(n)$.

Combining Theorem B.3, Lemma A.3, and Claim B.2, we can recover a solution of cost at most $\tilde{O}(k^\varepsilon)\mathsf{OPT}_{junc}$ for ONLINE PAIRWISE WEIGHTED SPANNER on G from a solution of ONLINE STEINER LABEL COVER on H. $\qquad\square$

B.2 Missing proof for Lemma 4.3

Lemma 4.3. *There exists a distance-preserving weighted junction tree J with density at most* $\mathsf{OPT}/\sqrt{k}$.

Proof. Let G^* (a subgraph of G) be the optimal pairwise weighted spanner solution with cost OPT. The proof proceeds by considering the following two cases: 1) there exists a vertex $r \in V$ that belongs to at least $\sqrt{k}$ $s_i \rightsquigarrow t_i$ paths of distance at most $Dist(s_i, t_i)$ in G^* for distinct i, and 2) there is no such vertex $r \in V$.

For the first case, we consider the union of the $s_i \rightsquigarrow t_i$ paths in G^*, each of distance at most $Dist(s_i, t_i)$, that passes through r. This subgraph in G^* contains an in-arborescence and an out-arborescence both rooted at r, whose union forms a distance-preserving weighted junction tree. This distance-preserving weighted junction tree has cost at most OPT and connects at least $\sqrt{k}$ terminal pairs, so its density is at most $\mathsf{OPT}/\sqrt{k}$.

For the second case, each vertex $r \in V$ appears in at most $\sqrt{k}$ $s_i \rightsquigarrow t_i$ paths in G^*. More specifically, each edge $e \in E$ also appears in at most $\sqrt{k}$ $s_i \rightsquigarrow t_i$ paths in G'. By creating $\sqrt{k}$ copies of each edge, all terminal pairs can be connected by edge-disjoint paths. Since the overall duplicate cost is at most $\sqrt{k} \cdot \mathsf{OPT}$, at least one of these paths has cost at most $\sqrt{k} \cdot \mathsf{OPT}/k$. This path constitutes a distance-preserving weighted junction tree whose density is at most $\mathsf{OPT}/\sqrt{k}$. $\qquad\square$