Online Directed Spanners and Steiner Forests

Elena Grigorescu
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Young-San Lin[1]
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Kent Quanrud
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Abstract

We present online algorithms for directed spanners and directed Steiner forests. These are well-studied network connectivity problems that fall under the unifying framework of online covering and packing linear programming formulations. This framework was developed in the seminal work of Buchbinder and Naor (Mathematics of Operations Research, 34, 2009) and is based on primal-dual techniques. Specifically, our results include the following:

- For the *pairwise spanner* problem, in which the pairs of vertices to be spanned arrive online, we present an efficient randomized algorithm with competitive ratio $\tilde{O}(n^{4/5})$ for graphs with general edge lengths, where n is the number of vertices of the given graph. For graphs with uniform edge lengths, we give an efficient randomized algorithm with competitive ratio $\tilde{O}(n^{2/3+\varepsilon})$, and an efficient deterministic algorithm with competitive ratio $\tilde{O}(k^{1/2+\varepsilon})$, where k is the number of terminal pairs. To the best of our knowledge, these are the first online algorithms for directed spanners. In the offline version, the current best approximation ratio for uniform edge lengths is $\tilde{O}(n^{3/5+\varepsilon})$, due to Chlamtáč, Dinitz, Kortsarz, and Laekhanukit (SODA 2017, TALG 2020).
- For the *directed Steiner forest* problem with uniform costs, in which the pairs of vertices to be connected arrive online, we present an efficient randomized algorithm with competitive ratio $\tilde{O}(n^{2/3+\varepsilon})$. The state-of-the-art online algorithm for general costs is due to Chakrabarty, Ene, Krishnaswamy, and Panigrahi (SICOMP 2018) and is $\tilde{O}(k^{1/2+\varepsilon})$-competitive. In the offline version, the current best approximation ratio with uniform costs is $\tilde{O}(n^{26/45+\varepsilon})$, due to Abboud and Bodwin (SODA 2018).

To obtain *efficient* and *competitive* online algorithms, we observe that a small modification of the online covering and packing framework by Buchbinder and Naor implies a polynomial-time implementation of the primal-dual approach with separation oracles, which a priori might perform exponentially many calls to the oracle. We convert the online spanner problem into an online covering problem and complete the rounding-step analysis in a problem-specific fashion.

2012 ACM Subject Classification Theory of computation → Online algorithms; Theory of computation → Packing and covering problems; Theory of computation → Routing and network design problems; Theory of computation → Rounding techniques

Keywords and phrases online directed pairwise spanners, online directed Steiner forests, online covering/packing linear programming, primal-dual approach

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2021.5

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2103.04543` [52]

Funding E.G and Y.L. were supported in part by NSF CCF-1910659 and NSF CCF-1910411.

Acknowledgements We thank the anonymous reviewers for comments and suggestions that helped improve the presentation. We thank Anupam Gupta and Greg Bodwin for bringing to our attention references that we missed in previous versions of the writeup.

[1] Corresponding author

1 Introduction

We study online variants of directed network optimization problems. In an online problem, the input is presented sequentially, one item at a time, and the algorithm is forced to make irrevocable decisions in each step, without knowledge of the remaining part of the input. The performance of the algorithm is measured by its *competitive ratio*, which is the ratio between the value of the online solution and that of an optimal offline solution.

Our main results focus on *directed spanners*, which are sparse subgraphs that approximately preserve pairwise distances between vertices. Spanners are fundamental combinatorial objects with a wide range of applications, such as distributed computation [9, 69], data structures [5, 75], routing schemes [35, 67, 70, 72], approximate shorthest paths [18, 41, 42], distance oracles [18, 31, 68], and property testing [8, 22]. For a comprehensive account of the literature, we refer the reader to the excellent survey [2].

We also study related network *connectivity* problems, and in particular on *directed Steiner forests*, which are sparse subgraphs that maintain connectivity between target terminal vertex pairs. Steiner forests are ubiquitously used in a heterogeneous collection of areas, such as multicommodity network design [49, 53], mechanism design and games [30, 63, 64, 73], computational biology [62, 71], and computational geometry [19, 24].

Our approaches are based on covering and packing linear programming (LP) formulations that fall into the unifying framework developed by Buchbinder and Naor [26], using the powerful primal-dual technique [51]. This unifying framework extends across widely different domains, and hence provides a general abstraction that captures the algorithmic essence of all online covering and packing formulations. In our case, to obtain *efficient* competitive algorithms for solving the LPs online, we observe that the algorithms in [26] can be slightly modified to significantly speed up the setting of our applications, in which the algorithm might otherwise make exponentially many calls to a separation oracle. This component is not tailored to the applications studied here and may be of independent interest. In particular, previous approaches solving online covering and packing problems either focus on the competitiveness of the algorithm [4, 12, 16], or manage to leverage the specific structure of the problem for better time efficiency in a somewhat ad-hoc manner [3, 11, 15, 21, 25, 59], while here the solution may be viewed as a more *unified* framework that is also *efficient*.

1.1 Our contributions

1.1.1 Directed spanners

Let $G = (V, E)$ be a directed simple graph with n vertices. Each edge is associated with its *length* $\ell : E \to \mathbb{R}_{\geq 0}$. The edge lengths are *uniform* if $\ell(e) = 1$ for all $e \in E$. In spanner problems, the goal is to compute a minimum cardinality (number of edges) subgraph in which the distance between terminals is preserved up to some prescribed factor. In the most well-studied setting, called the *directed s-spanner* problem, there is a fixed value $s \geq 1$ called the *stretch*, and the goal is to find a minimum cardinality subgraph in which *every* pair of vertices has distance within a factor of s in the original graph. For low stretch spanners, when $s = 2$, there is a tight $\Theta(\log n)$-approximation algorithm [44, 65]; when $s = 3, 4$ both with uniform edge lengths, there are $\tilde{O}(n^{1/3})$-approximation algorithms [20, 40]. When $s > 4$, the best known approximation factor is $\tilde{O}(n^{1/2})$ [20]. The problem is hard to approximate within an $O(2^{\log^{1-\varepsilon} n})$ factor for $3 \leq s = O(n^{1-\delta})$ and any $\varepsilon, \delta \in (0, 1)$, unless $NP \subseteq DTIME(n^{\text{polylog}\, n})$ [45].

A more general setting, called the *pairwise spanner* problem [34], and the *client-server* model [22, 44], considers an arbitrary set of terminals $D = \{(s_i, t_i) \mid i \in [k]\} \subseteq V \times V$. Each terminal pair (s_i, t_i) has its own target distance d_i. The goal is to compute a minimum cardinality subgraph in which for each i, the distance from s_i to t_i is at most d_i. For the pairwise spanner problem with uniform edge lengths, [34] obtains an $\tilde{O}(n^{3/5+\varepsilon})$-approximation.

In the online version, the graph is known ahead of time, and the terminal pairs and the corresponding target distances are received one by one in an online fashion. The distance requirement of the arriving terminal pair is satisfied by irrevocably including edges. There are no online algorithms for the pairwise spanner problem that we are aware of, even in the simpler and long-studied case of stretch s or graphs with uniform lengths.

For graphs with uniform edge lengths, we present the proof outline of the following theorem in Section 2 and refer the reader to the full version [52] for the complete proof.

▶ **Theorem 1.** *For the online pairwise spanner problem with uniform edge lengths, there exists a deterministic polynomial time algorithm with competitive ratio $\tilde{O}(k^{1/2+\delta})$ for any constant $\delta > 0$.*

Next, we turn to graphs with general edge lengths and derive online algorithms with competitive ratios in terms of n. We present a generic algorithm (Algorithm 3) used for Theorems 2, 3, 4, 5, 6, and 7. Due to space limitations, we refer the reader to the full version [52] for the complete proof of Theorems 3, 4, and 5.

For graphs with general edge lengths, we show the following in Section 3.1.

▶ **Theorem 2.** *For the online pairwise spanner problem, there is a randomized polynomial time algorithm with competitive ratio $\tilde{O}(n^{4/5})$.*

In one special case, the given graph might have uniform edge lengths, and the diameter is bounded or it is guaranteed that the distances between the terminal pairs are bounded. Let $d = \max_{i\in[k]}\{d_i\}$ that is known offline be the maximum allowed distance of any pair of terminals in the input. This setting is equivalent to the *d-diameter spanning subgraph* problem introduced in [22].

▶ **Theorem 3.** *For the online pairwise spanner problem with uniform edge lengths and maximum allowed distance d, there is a randomized polynomial time algorithm with competitive ratio $\tilde{O}(d^{1/3}n^{2/3})$.*

Another special case is where the edge lengths are *quasimetric*. That is, they satisfy the following directed form of the triangle inequality. For any two edges $u \to v$ and $v \to w$, there is also an edge $u \to w$ such that $\ell(u, w) \leq \ell(u, v) + \ell(v, w)$. This setting includes the class of transitive-closure graphs with uniform edge lengths, in which each pair or vertices connected by a directed path is also connected by a directed edge. The offline version of the *transitive-closure spanner* problem was formally defined in [22].

▶ **Theorem 4.** *For the online pairwise spanner problem where edge lengths are quasimetric, there is a randomized polynomial time algorithm with competitive ratio $\tilde{O}(n^{2/3})$.*

In a special case on graphs with uniform edge lengths, for each terminal pair (s_i, t_i), there is also an edge $s_i \to t_i$ in the given graph. This setting is equivalent to the *all-server spanner* problem introduced in [44].

▶ **Theorem 5.** *For the online all-server spanner problem with uniform edge lengths, there is a randomized polynomial time algorithm with competitive ratio $\tilde{O}(n^{2/3})$.*

For graphs with uniform edge lengths without further assumptions, we use Theorem 1 and the generic algorithm to prove the following theorem in Section 3.2.

▶ **Theorem 6.** *For the online pairwise spanner problem with uniform edge lengths, there is a randomized polynomial time algorithm with competitive ratio* $\tilde{O}(n^{2/3+\varepsilon})$ *for any constant* $\varepsilon \in (0, 1/3)$.

1.1.2 Directed Steiner forests

In the *directed Steiner forest* problem, we are given a directed graph $G = (V, E)$ with edge costs $w : E \to \mathbb{R}_{\geq 0}$, and a set of terminals $D = \{(s_i, t_i) \mid i \in [k]\} \subseteq V \times V$. The goal is to find a subgraph $H = (V, E')$ which includes an $s_i \rightsquigarrow t_i$ path for each terminal pair (s_i, t_i), and the total cost $\sum_{e \in E'} w(e)$ is minimized. The costs are *uniform* when $w(e) = 1$ for all $e \in E$.

In the offline setting with general costs, the best known approximations are $O(k^{1/2+\varepsilon})$ by Chekuri et al. [32] and $O(n^{2/3+\varepsilon})$ by Berman et al. [20]. For the special case of uniform costs, there is an improved approximation factor of $\tilde{O}(n^{26/45+\varepsilon})$ by Abboud and Bodwin [1]. In the online setting, Chakrabarty et al. [28] give an $\tilde{O}(k^{1/2+\varepsilon})$ approximation for general costs. Their algorithm also extends to the more general buy-at-bulk version. We prove the following in Section 3.3..

▶ **Theorem 7.** *For the online directed Steiner forest problem with uniform costs, there is a randomized polynomial time algorithm with competitive ratio* $\tilde{O}(n^{2/3+\varepsilon})$ *for any constant* $\varepsilon \in (0, 1/3)$.

We essentially improve the competitive ratio when the number of terminal pairs is $\omega(n^{4/3})$.

1.1.3 Summary

We summarize our main results for online pairwise spanners and directed Steiner forests in Table 1 by listing the competitive ratios and contrast them with the corresponding known competitive and approximation ratios. We note that offline $\tilde{O}(n^{4/5})$-approximate pairwise spanners for graphs with general edge lengths and offline $\tilde{O}(k^{1/2+\varepsilon})$-approximate pairwise spanners for graphs with uniform edge lengths can be obtained by our online algorithms.

Table 1 Summary of the competitive and approximation ratios. Here, n refers to the number of vertices and k refers to the number of terminal pairs. We include the known results for comparison. The text in gray refers to known results while the text in black refers to our contributions.

Setting	Pairwise Spanners	Directed Steiner Forests
Offline	$\tilde{O}(n^{4/5})$ (implied by Thm 2)	$\tilde{O}(n^{26/45+\varepsilon})$ (uniform costs) [1]
	$\tilde{O}(n^{3/5+\varepsilon})$ (uniform lengths) [34]	$O(n^{2/3+\varepsilon})$ [20]
	$\tilde{O}(k^{1/2+\varepsilon})$ (uniform lengths, implied by Thm 1)	$O(k^{1/2+\varepsilon})$ [32]
Online	$\tilde{O}(n^{4/5})$ (Thm 2)	$\tilde{O}(k^{1/2+\varepsilon})$ [28]
	$\tilde{O}(n^{2/3+\varepsilon})$ (uniform lengths, Thm 6)	$\tilde{O}(n^{2/3+\varepsilon})$ (uniform costs, Thm 7)
	$\tilde{O}(k^{1/2+\varepsilon})$ (uniform lengths, Thm 1)	

1.2 An efficient online covering and packing framework

Before presenting our modification to the unified framework in [26] to obtain efficient online covering and packing LP solvers, we give an overview of the well-known primal-dual framework

for approximating covering and packing LP's online. This framework is the main engine of our application and it is important to establish some context before getting into the application for spanners and Steiner forests. We also introduce a discussion of certain technical nuances that arise for our application, and the small modification we propose to address it. A more formal description, including proofs and fully parameterized theorem statements, is fairly technical and therefore deferred to Appendix C *after* we have used these tools in the context of spanners and Steiner forests.

The primal-dual framework was first developed for the online set cover problem in the seminal work of [4]. The approach was extended to network optimization problems in undirected graphs in [3], then abstracted and generalized to a broad LP-based primal-dual framework in [26]. Our discussion primarily centers around the abstract framework in [26]. A number of previous results in online algorithms, such as ski rental [61] and paging [16], can be recovered from this approach and many new important applications have since been developed, such as the k-server problem [76]. We refer the reader to the excellent survey by Buchbinder and Naor [27].

These works develop a clean two-step approach to online algorithms based on 1) solving the LP online, and 2) rounding the LP online. Solving the LP online can be done in a generic fashion, while rounding tends to be problem-specific. The setting for the *covering* LP is the following.

$$\text{minimize } \langle \mathbf{c}, x \rangle \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ s.t. } Ax \geq \mathbf{b}. \tag{1}$$

Here $A \in \mathbb{R}^{m \times n}_{\geq 0}$ consists of m covering constraints, $\mathbf{b} \in \mathbb{R}^n_{>0}$ is a positive lower bound of the covering constraints, and $\mathbf{c} \in \mathbb{R}^m_{>0}$ denotes the positive coefficients of the linear cost function. Each constraint can be normalized, so we focus on covering LP's in the following form.

$$\text{minimize } \langle \mathbf{c}, x \rangle \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ s.t. } Ax \geq \mathbf{1} \tag{2}$$

where $\mathbf{1}$ is a vector of all ones.

In the online covering problem, the cost vector $\mathbf{c}$ is given offline, and each of these covering constraints is presented one by one in an online fashion, that is, m can be unknown. The goal is to update x in a non-decreasing manner such that all the covering constraints are satisfied and the objective value $\langle \mathbf{c}, x \rangle$ is approximately optimal. An important idea in this line of work is to simultaneously consider the dual *packing* problem:

$$\text{maximize } \langle \mathbf{1}, y \rangle \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ s.t. } A^T y \leq \mathbf{c} \tag{3}$$

where A^T consists of n packing constraints with an upper bound $\mathbf{c}$ given offline.

In the online packing problem, the *columns* of A^T and the corresponding variables are presented online with value zero, one can either let the arriving variable remain zero, or irrevocably assign a positive value to the arriving variable. The goal is to approximately maximize the objective value $\langle \mathbf{1}, y \rangle$ with each constraint approximately satisfied.

1.2.1 Separation oracles in the online setting

The primal-dual framework in [26] simultaneously solves both LP (2) and LP (3), and crucially uses LP-duality and strong connections between the two solutions to argue that they are both nearly optimal. Here we give a sketch of the LP solving framework for reference in the subsequent discussion. We maintain solutions x and y for LP (2) and LP (3), respectively, in an online fashion. The covering solution x is a function of the packing solution y. In particular, each coordinate x_j is exponential in the *load* of the corresponding packing constraint in LP

(3). Both x and y are monotonically increasing. The algorithm runs in phases, where each phase corresponds to an estimate for OPT revised over time. Within a phase we have the following. If the new covering constraint $i \in [m]$, presented online, is already satisfied, then there is nothing to be done. Otherwise, increase the corresponding coordinate y_i, which simultaneously increases the x_j's based on the magnitude of the coordinate a_{ij}, where a_{ij} is the i-th row j-th column entry of A. The framework in [26] increases y_i until the increased x_j's satisfy the new constraint. This naturally extends to the setting when the problem relies on a *separation oracle* to retrieve an unsatisfied covering constraint where the number of constraints can be unbounded [26]. However, while this approach will fix all violating constraints, each individual fix may require a diminishingly small adjustment that cannot be charged off from a global perspective. Consequently the algorithm may have to address exponentially many constraints.

1.2.2 A primal-dual bound on separation oracles

Our goal is to adjust the framework to ensure that we only address a polynomial number of constraints (per phase). For many concrete problems in the literature, this issue can be addressed directly based on the problem at hand (discussed in greater detail in Section A). In our setting, we start with a combinatorially defined LP that is not a pure covering problem, and convert it to a covering LP. While having a covering LP is conducive to the online LP framework, the machinery generates a large number of covering constraints that are very unstructured. For example, we have little control over the coefficients of these constraints. This motivates us to develop a more generic argument to bound the number of queries to the separation oracle, based on the online LP framework, more so than the exact problem at hand. Here, when addressing a violated constraint i, we instead increase the dual variable y_i until the increased primal variables x (over-)satisfy the new constraint by a factor of 2. This forces at least one x_j to be doubled – and in the dual, this means we used up a substantial amount of the corresponding packing constraint. Since the packing solution is already guaranteed to be feasible in each phase by the overall framework, this leads us to conclude that we only ever encounter polynomially many violating constraints.

For our modified online covering and packing framework, we show that 1) the approximation guarantees are identical to those in [26], 2) the framework only encounters polynomially many violating constraints for the online covering problem, and 3) only polynomially many updates are needed for the online packing problem.

▶ **Theorem 8** (Informal). *There exists an $O(\log n)$-competitive online algorithm for the covering LP (2) which encounters polynomially many violating constraints.*

▶ **Theorem 9** (Informal). *Given any parameter $B > 0$, there exists a $1/B$-competitive online algorithm for the packing LP (3) which updates y polynomially many times, and each constraint is violated within an $O(f(A)/B)$ factor ($f(A)$ is a logarithmic function that depends on the entries in A).*

We note that the competitive ratios given in [26] are tight, which also implies the tightness of the modified framework. The number of violating constraints depends not only on the number of covering variables and packing constraints n, but also on the number of bits used to present the entries in A and $\mathbf{c}$. The formal proof for Theorem 8 is provided in Appendix C, while the formal proof for Theorem 9 provided in the full version [52] is not directly relevant to this work, but may be of independent interest.

1.3 High-level technical overview for online network optimization problems

1.3.1 Online pairwise spanners

For this problem, a natural starting point is the flow-based LP approach for offline s-stretch directed spanners, introduced in [38]. The results of [34] adopt a slight tweak for this approach to achieve an $\tilde{O}(n/\sqrt{\mathsf{OPT}})$-approximation, where OPT is the size of the optimal solution. With additional ideas, the $\tilde{O}(n/\sqrt{\mathsf{OPT}})$-approximation is converted into an $\tilde{O}(n^{3/5+\varepsilon})$-approximation for pairwise spanners. One technical obstacle in the online setting is the lack of a useful lower bound for OPT. Another challenge is solving the LP for the spanner problem and rounding the solution in an online fashion, particularly as the natural LP is not a covering LP. We address these technical obstacles as discussed below in Section 3. Ultimately we obtain an $\tilde{O}(n^{4/5})$ competitive ratio for the online setting. The strategy here is to convert the LP for spanners into a covering LP, where the constraints are generated by an *internal* LP. The covering LP previously appeared in [38] implicitly, and in [39] explicitly.

1.3.2 Online pairwise spanners with uniform edge lengths

For the special case of uniform edge lengths, [34] obtains an improved bound of $\tilde{O}(n^{3/5+\varepsilon})$. It is natural to ask if the online bound of $\tilde{O}(n^{4/5})$ mentioned above can be improved as well. Indeed, we obtain an improved bound of $\tilde{O}(n^{2/3+\varepsilon})$ by replacing the greedy approach in the small OPT regime by using the $\tilde{O}(k^{1/2+\varepsilon})$-competitive online algorithm discussed in Section 2. This algorithm leverages ideas from [34] in reducing to label cover problems with ideas from the online network design algorithms of [28]. Some additional ideas are required to combine the existing tools and among others we had to formulate a new pure covering LP that can be solved online, to facilitate the transition.

1.3.3 Online Steiner forests with uniform costs

This problem is a special case of the online pairwise spanner problem where the distance requirement for each terminal pair is infinity and the edge lengths are uniform. The online algorithm for this problem has a similar structure to the one for pairwise spanners and similar obstacles to overcome. Before small value of OPT gets large, we can leverage the $\tilde{O}(k^{1/2+\varepsilon})$-competitive online algorithm or the online buy-at-bulk framework [28] for a better bound than a greedy approach would give, improving the competitive ratio to $\tilde{O}(n^{2/3+\varepsilon})$.

1.4 Organization

Since the proof of Theorem 1 is the most involved contribution of this work, we start by presenting the proof outline in Section 2 and refer the reader to the full version [52] for the technical proof details. In Section 3, we prove Theorems 2, 6, and 7 by designing a generic online algorithm, which is also used for proving Theorems 3, 4, and 5 in the full version [52]. We show the modified online covering framework in Appendix C, while the modified online packing framework is presented in the full version [52]. We refer the reader to Appendix A for a detailed description of related work and an exposition situating our work in the expansive literature of directed spanners, Steiner forests, and covering and packing problems.

2 Online Pairwise Spanners with Uniform Lengths

In this section, we present the proof outline of Theorem 1, namely we design an online algorithm for the pairwise spanner problem with uniform edge lengths with competitive ratio $\tilde{O}(k^{1/2+\delta})$ for any constant $\delta > 0$. We recall that in the *pairwise spanner* problem, we are given a directed graph $G = (V, E)$ with edge length $\ell : E \to \mathbb{R}_{\geq 0}$, a general set of k terminals $D = \{(s_i, t_i) \mid i \in [k]\} \subseteq V \times V$, and a target distance d_i for each terminal pair (s_i, t_i), the goal is to output a subgraph $H = (V, E')$ of G such that for every pair $(s_i, t_i) \in D$ it is the case that $d_H(s_i, t_i) \leq d_i$, i.e. the length of a shortest $s_i \rightsquigarrow t_i$ path is at most d_i in the subgraph H, and we want to minimize the number of edges in E'. The edge lengths are uniform if $\ell(e) = 1$ for all $e \in E$. In the online setting, the directed graph G is given offline, while the vertex pairs in $D \subseteq V \times V$ arrive online one at a time. In the beginning, $E' = \emptyset$. Suppose (s_i, t_i) and its target distance d_i arrive in round i, we select some edges from E and irrevocably add them to E', such that in the subgraph $H = (V, E')$, $d_H(s_i, t_i) \leq d_i$.

2.1 Outline of the proof of Theorem 1

We start by describing the high-level approach of our proof of Theorem 1. While the proof combines ideas of the online buy-at-bulk framework in [28] and of the reduction from the pairwise spanner problem to a connectivity problem in [34], implementing the details require several new ideas. Specifically, we introduce a useful extension of the Steiner problem, called the *Steiner label cover* problem, and our main contribution is an online covering LP formulation for this problem. This approach allows us to not only capture the global approximation property in an online setting, as in [28], but also to handle distance constraints, as in [34]. The entire proof consists of three main ingredients:

1. We first show that there exists an $O(\sqrt{k})$-approximate solution consisting of *junction trees*. A junction tree is a subgraph consisting of an in-arborescence and out-arborescence rooted at the same vertex (see also Definition 10).
2. We then show a reduction from the online pairwise spanner problem to the online Steiner label cover problem on a forest with a loss of an $O(k^{1/2+\delta})$ factor. More precisely, an $O(\sqrt{k})$ factor comes from the junction tree approximation and an extra $O(k^{\delta})$ factor comes from the *height reduction* technique introduced in [32, 58]. The height reduction technique allows us to focus on low-cost trees of height $O(1/\delta)$ in order to recover a junction tree approximation.
3. Finally, we show a reduction from the online Steiner label cover problem to the online undirected Steiner forest problem, with a loss of a $\mathsf{polylog}(n)$ factor. More precisely, we first formulate an online covering LP for the online Steiner label cover instance, then construct an online undirected Steiner forest instance from the LP solution, with a loss of a factor of 2. By [21], the online undirected Steiner forest problem can then be solved deterministically with competitive ratio $\mathsf{polylog}(n)$.

Combining these three ingredients results in an $\tilde{O}(k^{1/2+\delta})$-competitive algorithm. We provide further intuition below. The detailed description these three ingredients are presented in the full version [52].

2.1.1 Junction tree approximation

Many connectivity problems, including Steiner forests, buy-at-bulk, and spanner problems, are usually solved using *junction trees* introduced in [33].

▶ **Definition 10.** *A* junction tree *rooted at $r \in V$ is a directed graph $G = (V, E)$, by taking the union of an in-arborescence rooted at r and an out-arborescence rooted at r*[2]*. A* junction tree solution *is a collection of junction trees rooted at different vertices, that satisfies all the terminal distance constraints.*

▶ **Lemma 11.** *There exists an $O(\sqrt{k})$-approximate junction tree solution for pairwise spanners.*

At a high level, the proof of Lemma 11 follows by a standard *density* argument. A *partial solution* is a subgraph that connects a subset of the terminal pairs within the required distances. The density of a partial solution is the ratio between the number of edges used and the number of terminal pairs connected within the required distances. This argument is used for solving offline problems including the Steiner forest problem [20, 32, 46], the buy-at-bulk problem [6], the Client-Server s-spanner [22] problem, and the pairwise spanner problem [34], by greedily removing low density partial solutions in an iterative manner. Fortunately, this iterative approach also guarantees a nice *global* approximation that consists of junction trees rooted at different vertices, which is amenable in the online setting. The online buy-at-bulk algorithm in [28] is an online version of the junction tree framework for connectivity problems. Our main technical contribution is further modifying the online version of the junction tree framework for problems with distance constraints.

2.1.2 Reduction to Steiner label cover

We reduce the pairwise spanner problem to the following extension of Steiner problem termed *Steiner label cover*.

▶ **Definition 12.** *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.*

For the online Steiner label cover problem, (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.

To reduce to the online Steiner label cover problem, we construct a directed graph G' that consists of disjoint layered graphs from the given graph $G = (V, E)$. Each vertex in G' is labelled by the distance to (from) the root of a junction tree. This allows us to capture distance constraints by a Steiner label cover instance with distance-based relations. From G', we further construct an undirected graph H which is a forest by the *height reduction* technique [32, 58]. In H, we define the corresponding Steiner label cover instance, where the terminal vertex sets consist of the leaves, and the solution is guaranteed to be a forest. The Steiner label cover instance on the forest H has a nice property. For each tree in H, the terminal vertices can be ordered in a way such that if an *interval* belongs to the relation, then any *subinterval* also belongs to the relation.

[2] A junction tree does not necessarily have a tree structure in directed graphs, i.e. edges in the in-arborescence and edges in the out-arborescence may overlap. Nevertheless, we continue using this term because of historical reasons. A similar notion can also be used for undirected graphs, where a junction tree is indeed a tree.

▶ **Definition 13.** *The* ordered Steiner label cover problem on a forest *is defined as a special case of the Steiner label cover problem (see Definition 12) with the following properties.*

1. *G is an undirected graph consisting of disjoint union of trees $H_1, H_2, \ldots, H_n$ each of which has a distinguished root vertex r_j where $j \in [n]$.*
2. *For each (S_i, T_i) and R_i, and each tree H_j, the input also includes the orderings $\prec_{i,j}$ such that:*
   a. *For $S_i^j := S_i \cap H_j$ and $T_i^j := T_i \cap H_j$, the ordering $\prec_{i,j}$ is defined on $S_i^j \cup T_i^j$.*
   b. *The root r_j separates S_i^j from T_i^j.*
   c. *If $s \in S_i^j$ and $t \in T_i^j$ are such that $(s,t) \in R_i$, then for any $s' \in S_i^j$ and $t' \in T_i^j$ such that $s \preceq_{i,j} s' \prec_{i,j} t' \preceq_{i,j} t$, we have that $(s', t') \in R_i$.*

We note that for the online ordered Steiner label cover problem on a forest, besides (S_i, T_i) and R_i, the orderings $\{\succ_{i,j}\}_{j \in [n]}$ also arrive online.

We employ a well-defined mapping between junction trees in G and forests in H by paying an $\tilde{O}(k^{1/2+\delta})$ factor for competitive online solutions. A crucial step for showing Theorem 1 is the following theorem.

▶ **Theorem 14.** *For any constant $\delta > 0$, an α-competitive polynomial time algorithm for online ordered Steiner label cover on a forest implies an $O(\alpha k^{1/2+\delta})$-competitive polynomial time algorithm for the online pairwise spanner problem on a directed graph with uniform edge lengths.*

At a high level, the online pairwise spanner problem on a directed graph $G = (V, E)$ with uniform edge lengths reduces to an instance of online Steiner label cover on the forest H with the following properties.

1. H consists of disjoint trees H_r rooted at r' for each vertex $r \in V$.
2. $|V(H)| = n^{O(1/\delta)}$, $E(H) = n^{O(1/\delta)}$, and each tree H_r has depth $O(1/\delta)$ with respect to r'.
3. For each arriving terminal pair (s_i, t_i) with distance requirement d_i, there is a corresponding pair of terminal sets $(\hat{S}_i, \hat{T}_i)$ and relation $\hat{R}_i$ with $|\hat{R}_i| = n^{O(1/\delta)}$, where $\hat{S}_i$ and $\hat{T}_i$ are disjoint subsets of leaves in H. Furthermore, we can generate orderings $\prec_{i,r}$ based on the distance-based relations $\hat{R}_i$ such that the Steiner label cover instance is an ordered instance on the forest H.

This technique closely follows the one for solving offline pairwise spanners in [34]. The intermediary problem considered in [34] is the *minimum density Steiner label cover problem*. In this framework, the solution is obtained by selecting the partial solution with the lowest density among the junction trees rooted at different vertices and repeat. In the online setting, to capture the global approximation for pairwise spanners, we construct a forest H and consider all the possible roots simultaneously.

2.1.3 An online algorithm for Steiner label cover on H

The goal is to prove the following lemma.

▶ **Lemma 15.** *For the online ordered Steiner label cover problem on a forest (see definition 13), there is a deterministic polynomial time algorithm with competitive ratio* $\mathsf{polylog}(n)$.

We derive an LP formulation for the Steiner label cover instance on H. At a high level, the LP minimizes the total edge weight by selecting edges that cover paths with endpoint pairs which belong to the distance-based relation. We show that the LP for Steiner label cover can be converted into an online covering problem, which is efficiently solvable by Theorem 8.

The online rounding is based on the online LP solution for Steiner label cover. Given the online LP solution and the orderings in round i generated by the distance-based relation $\hat{R}_i$, we extract the representative vertex sets $\tilde{S}_i$ and $\tilde{T}_i$ from the terminal sets $\hat{S}_i$ and $\hat{T}_i$, respectively, according to orderings $\prec_{i,r}$ and the contribution of the terminal vertex to the objective of the Steiner label cover LP. We show that the union of cross-products over partitions of $\tilde{S}_i$ and $\tilde{T}_i$ (based on the trees in H) is a subset of the distance-based relation $\hat{R}_i$. This allows us to reduce the online ordered Steiner label cover problem to the online undirected Steiner forest problem by connecting a super source to $\tilde{S}_i$ and a super sink to $\tilde{T}_i$.

This technique closely follows the one for solving offline pairwise spanners in [34]. The main difference is that in the offline pairwise spanner framework, the LP formulation is density-based and considers only one (fractional) junction tree. To globally approximate the online pairwise spanner solution, our LP formulation is based on the forest H and its objective is the total weight of a (fractional) forest.

The LP for the undirected Steiner forest problem is roughly in the following form.

$$\begin{aligned} \min_{x} \quad & \sum_{e \in E(H)} w'(e) x_e & & \\ \text{subject to} \quad & x \text{ supports an } \tilde{S}_i\text{-}\tilde{T}_i \text{ flow of value 1} & & \forall i \in [k], \\ & x_e \geq 0 & & \forall e \in E(H). \end{aligned} \tag{4}$$

Here w' denotes the edge weights in H. We show that a solution of the undirected Steiner forest LP (4) recovers a solution for the Steiner label cover LP by a factor of 2. The integrality gap of the undirected Steiner forest LP is $\mathsf{polylog}(n)$ because the instance can be decomposed into single source Steiner forest instances by the structure of H [28, 50]. This implies that the online rounding for the Steiner label cover LP can be naturally done via solving the undirected Steiner forest instance online, by using the $\mathsf{polylog}(n)$-competitive framework [21].

2.1.3.1 Putting it all together

We summarize the overall $\tilde{O}(k^{1/2+\delta})$-competitive algorithm for online pairwise spanners when the given graph has uniform edge lengths. The reduction strategy is as follows:

1. Reduce the online pairwise spanner problem on the original graph G to the online Steiner label cover problem on the directed graph G' which consists of disjoint layered graphs.
2. Reduce the online Steiner label cover problem on G' to an online ordered Steiner label cover problem on H, where H is a forest.
3. In the forest H, reduce the online ordered Steiner label cover problem to the online undirected Steiner forest problem.

We note that G' and H are constructed offline, while the graph for the final undirected Steiner forest instance is partially constructed online, by adding super sources and sinks and connecting incident edges to the representative leaf vertices online in H. The pairwise spanner in G is $O(\sqrt{k})$-approximated by junction trees according to Lemma 11. The graph G' preserves the same cost of the pairwise spanner (junction tree solution) in G. The solution of the ordered Steiner label cover problem in graph H is a forest. One can map a forest in H to junction trees in G', via the height reduction technique by losing an $O(k^{\delta})$ factor. Finally, in the forest H, we solve the undirected Steiner forest instance online and recover an ordered Steiner label cover solution by losing a $\mathsf{polylog}(n)$ factor. The overall competitive ratio is therefore $\tilde{O}(k^{1/2+\delta})$.

3 Online Pairwise Spanners

We recall that in the general *pairwise spanner* problem, we are given a directed graph $G = (V, E)$ with edge length $\ell : E \to \mathbb{R}_{\geq 0}$, a general set of k terminals $D = \{(s_i, t_i) \mid i \in [k]\} \subseteq V \times V$, and a target distance d_i for each terminal pair (s_i, t_i), the goal is to output a subgraph $H = (V, E')$ of G such that for every pair $(s_i, t_i) \in D$ it is the case that $d_H(s_i, t_i) \leq d_i$, i.e. the length of a shortest $s_i \rightsquigarrow t_i$ path is at most d_i in the subgraph H, and we want to minimize the number of edges in E'.

3.1 An $\tilde{O}(n^{4/5})$-competitive online algorithm for pairwise spanners

In this section, we prove Theorem 2. Recall that in the online setting of the problem, the directed graph G is given offline, while the vertex pairs in $D \subseteq V \times V$ arrive online one at a time. In the beginning, $E' = \emptyset$. Suppose (s_i, t_i) and its target distance d_i arrive in round i, we select some edges from E and irrevocably add them to E', such that in the subgraph $H = (V, E')$, $d_H(s_i, t_i) \leq d_i$. The goal is to approximately minimize the total number of edges added to E'.

We start with a high-level sketch of an offline algorithm, which we will build on for the online setting. The randomized rounding framework in [20, 34] has two main steps. One step is to solve and round an LP for the spanner problem. The second is to uniformly sample vertices and add the shortest path in-arboresences and out-arboresences rooted at each of the sampled vertices. Terminal pairs are classified as either *thin* or *thick* and are addressed by one of the two steps above accordingly.

In the first step, the rounding scheme based on an LP solution for spanners ensures with high probability that for all thin terminal pairs the distance requirements are met. The second step ensures with high probability that for all thick terminal pairs the distance requirements are met. By selecting an appropriate threshold for classifying the thin and thick pairs, this leads to an $O(n/\sqrt{\mathsf{OPT}})$-approximation, where OPT is the number of edges in the optimal solution.

The main challenges in adapting this approach to the online setting are as follows: 1) OPT can be very small, and 2) the LP for spanners is not naturally a pure covering LP, which makes it difficult to solve online. In the previous work in the offline setting, the small-OPT case is addressed by sophisticated strategies that appear difficult to emulate online. Instead, we show that the optimal value (however small) is at least the square root of the number of terminal pairs that have arrived. Thus, if OPT is small and not many pairs have arrived, we can greedily add a path with the fewest edges subject to the distance requirement for each pair. To overcome the second challenge, we convert the LP for spanners into an equivalent covering LP as in [39], where exponentially many covering constraints are generated by an auxiliary LP. Having transformed the LP into a purely covering one, we can solve the LP online, treating the auxiliary LP as a separation oracle.

3.1.1 A simple $\tilde{O}(n^{4/5})$-approximate offline algorithm based on [34]

For ease of exposition, we first design a simpler offline algorithm (slightly weaker than the state-of-the-art) that is more amenable to the online setting. This allows us to establish the main ingredients governing the approximation factor in a simpler setting, and then address the online aspects separately. The algorithm leverages the framework developed in [20, 34].

Let $\mathcal{P}_i$ denote the collection of $s_i \rightsquigarrow t_i$ paths of length at most d_i consisting of edges in E. Let the *local graph* $G^i = (V^i, E^i)$ be the union of all vertices and edges in $\mathcal{P}_i$. A pair $(s_i, t_i) \in D$ is *t-thick* if $|V^i| \geq t$, otherwise (s_i, t_i) is *t-thin*. Consider the following standard LP relaxation (essentially the one in [38]).

$$\begin{aligned} \min_{x,y} \quad & \sum_{e \in E} x_e \\ \text{subject to} \quad & \sum_{P \in \mathcal{P}_i} y_P \geq 1 && \forall i \in [k], \\ & \sum_{P | e \in P \in \mathcal{P}_i} y_P \leq x_e && \forall e \in E, \forall i \in [k], \\ & x_e \geq 0 && \forall e \in E, \\ & y_P \geq 0 && \forall P \in \mathcal{P}_i \quad \forall i \in [k]. \end{aligned} \tag{5}$$

Herein, x_e is an indicator of edge e and y_P is an indicator of path P. Suppose we have an integral feasible solution. Then the first set of constraints ensures that there is at least one $s_i \rightsquigarrow t_i$ path of length at most d_i selected, and the second set of constraints ensures that if a path P is selected, then all its edges are selected.

We say that a pair $(s_i, t_i) \in D$ is *settled* if the selection of edges is such that there exists an $s_i \rightsquigarrow t_i$ path of length at most d_i. Applying a simple rounding scheme based on a solution of LP (5) settles the thin pairs with high probability, while sampling enough vertices and adding shortest path in-arborescences and out-arborescences rooted at each sampled vertex ensures with high probability that thick pairs are settled. Let OPT be the optimum value of the given pairwise spanner instance. Without loss of generality, we may assume that we know OPT since we can guess every value of OPT in $[|E|]$ in the offline setting. Now we are ready to describe Algorithm 1 in [34].

▶ **Lemma 16.** ([34]) *Algorithm 1 is $\tilde{O}(n/\sqrt{OPT})$-approximate.*

Algorithm 1 Offline pairwise spanner.

1: $E' \leftarrow \emptyset$ and $t \leftarrow n/\sqrt{\mathsf{OPT}}$.
2: Solve LP (5) and add each edge $e \in E$ to E' with probability $\min\{1, x_e t \ln n\}$ independently.
3: Obtain a vertex set $W \subseteq V$ by sampling $(3n \ln n)/t$ vertices from V independently and uniformly at random. Add the edges of shortest path in-arborescences and out-arborescences rooted at w for each $w \in W$.

In the all-pairs spanner problem where OPT is $\Omega(n)$, Algorithm 1 is $\tilde{O}(\sqrt{n})$-approximate which matches the state-of-the-art approximation ratio given in [20]. For the pairwise spanner problem, the main challenge is the lack of a nice lower bound for OPT. In the offline setting, [34] achieves an $\tilde{O}(n^{3/5+\varepsilon})$-approximate solution by a careful case analysis when edges have uniform lengths. We give an alternative approach that is amenable to the online setting by considering two cases, where one resolves the issue when OPT does not have a nice lower bound, and the other uses a variant of Algorithm 1 given that OPT has a nice lower bound. This approach relies on the following observation.

▶ **Lemma 17.** $OPT \geq \sqrt{k}$.

Proof. We observe that when the spanner has ℓ edges, there are at most ℓ source vertices and ℓ sink vertices, so there are at most ℓ^2 terminal pairs. Therefore, when the spanner has OPT edges, there are at most OPT^2 terminal pairs, so $\mathsf{OPT} \geq \sqrt{k}$. ◀

Now we specify the simple offline algorithm given in Algorithm 2. In the beginning, we set two parameters T and t (which we will describe later), and set $E' = \emptyset$. An $s_i \rightsquigarrow t_i$ path is *cheapest feasible* if it meets the distance requirement d_i by using the minimum number of edges from E. We note that cheapest feasible paths can be found by Bellman-Ford algorithm.

Algorithm 2 Simple offline pairwise spanner.

1: **if** $k < T$ **then**
2: Add the edges of a cheapest feasible $s_i \rightsquigarrow t_i$ path to E' for each $i \in [k]$.
3: **else**
4: Solve LP (5) and add each edge $e \in E$ to E' with probability $\min\{1, x_e t \ln n\}$ independently.
5: Obtain a vertex set $W \subseteq V$ by sampling $(3n \ln n)/t$ vertices from V independently and uniformly at random. Add the edges of shortest path in-arborescences and out-arborescences rooted at each vertex $w \in W$ to E'.

▶ **Lemma 18.** *Algorithm 2 is $\tilde{O}(n^{4/5})$-approximate when $T = t = n^{4/5}$.*

Proof. If $k < n^{4/5}$, we add the edges of a cheapest feasible $s_i \rightsquigarrow t_i$ path for each $(s_i, t_i) \in D$. Each cheapest feasible $s_i \rightsquigarrow t_i$ path contains at most OPT edges, so the ratio between this solution and OPT is $n^{4/5}$. If $k \geq n^{4/5}$, then $\mathsf{OPT} \geq n^{2/5}$ by Lemma 17. Let LP^* be the optimal objective value of LP (5). The approximation guarantee is

$$\frac{\tilde{O}(t\mathsf{LP}^*) + \tilde{O}(n^2/t)}{\mathsf{OPT}} \leq \frac{\tilde{O}(n^{4/5}\mathsf{OPT}) + \tilde{O}(n^{6/5})}{\mathsf{OPT}} = \tilde{O}(n^{4/5})$$

since the number of edges retained from the rounding scheme is at most $\tilde{O}(t)\mathsf{LP}^*$ and the number of edges retained by adding arborescences is at most $2n \cdot 3n \ln n/t$. This summarizes the simple offline $\tilde{O}(n^{4/5})$-approximation algorithm. ◀

3.1.2 An $\tilde{O}(n^{4/5})$-competitive online algorithm

It remains to convert the simple offline algorithm to an online algorithm. We address the two main modifications.

1. We have to (approximately) solve LP (5) online, which is not presented as a covering LP.
2. We have to round the solution of LP (5) online.

For the first modification, LP (5) is converted to an equivalent covering LP (9) (which we show in Appendix B) and approximately solved in an online fashion. For the second modification, we use an online version of the rounding scheme in Algorithm 2, such that the overall probability (from round 1 to the current round) for the edge selection is consistent with the probability based on the online solution of LP (5), by properly scaling the probability based on a conditional argument.

The online algorithm in round i is given in Algorithm 3. The same structure is used for other variants of the online pairwise spanner problem. In the beginning, we pick a threshold parameter T and a thickness parameter t, and set $E' = \emptyset$. Let x_e^i denote the value of x_e in the approximate solution of LP (9) obtained in round i. Let $p_e^i := \min\{1, x_e^i t \ln n\}$. Algorithm 3 is the online version of Algorithm 2. A key insight is that when we add the arborescences in round T, it also settles the *future* thick terminal pairs with high probability. With the outline structure of the online algorithm, we prove Theorem 2 in Appendix B.

▶ **Theorem 2.** *For the online pairwise spanner problem, there is a randomized polynomial time algorithm with competitive ratio* $\tilde{O}(n^{4/5})$.

Algorithm 3 Online pairwise spanner.

for an arriving pair (s_i, t_i) **do**
  Convert the spanner LP (5) to the covering LP (9) and solve LP (9) online.
  if $i < T$ **then**
    Add the edges of a cheapest feasible $s_i \rightsquigarrow t_i$ path to E'.
  else if $i = T$ **then**
    Obtain a vertex set $W \subseteq V$ by sampling $(3n \ln n)/t$ vertices from V independently and uniformly at random. Add the edges of shortest path in-arborescences and out-arborescences rooted at each vertex $w \in W$ to E'.
    Add each edge e to E' independently with probability p_e^i for each edge $e \in E \setminus E'$.
  else ▷ $i > T$
    Add each edge e to E' independently with probability $(p_e^i - p_e^{i-1})/(1 - p_e^{i-1})$ for each edge $e \in E \setminus E'$.

3.2 Online pairwise spanners with uniform edge lengths

In this section, we prove Theorem 6.

▶ **Theorem 6.** *For the online pairwise spanner problem with uniform edge lengths, there is a randomized polynomial time algorithm with competitive ratio* $\tilde{O}(n^{2/3+\varepsilon})$ *for any constant* $\varepsilon \in (0, 1/3)$.

Proof. We employ Algorithm 3 with a slight tweak and set $T = \lfloor n^{4/3-4\varepsilon} \rfloor$ and $t = n^{2/3+\varepsilon}$.

If $k < T$, instead of adding edges of a shortest $s_i \rightsquigarrow t_i$ path, we use Theorem 1 to find an $\tilde{O}(n^{2/3+\varepsilon})$-competitive solution.

▶ **Theorem 1.** *For the online pairwise spanner problem with uniform edge lengths, there exists a deterministic polynomial time algorithm with competitive ratio* $\tilde{O}(k^{1/2+\delta})$ *for any constant* $\delta > 0$.

For any $\varepsilon \in (0, 1/3)$, there exists δ such that $4\delta/(9 + 12\delta) = \varepsilon$. By picking this δ, we have

$$\begin{aligned}(\frac{4}{3} - 4\varepsilon)(\frac{1}{2} + \delta) = (\frac{4}{3} - \frac{16\delta}{9+12\delta})(\frac{1}{2} + \delta) &= \frac{2}{3} + \frac{4\delta}{3} - \frac{8\delta + 16\delta^2}{9+12\delta} \\ &= \frac{2}{3} + \frac{12\delta + 16\delta^2 - 8\delta - 16\delta^2}{9+12\delta} = \frac{2}{3} + \frac{4\delta}{9+12\delta} = \frac{2}{3} + \varepsilon.\end{aligned}$$

Hence, the ratio between the solution obtained by Theorem 1 and OPT is

$$k^{1/2+\delta} \leq n^{(4/3-4\varepsilon)(1/2+\delta)} = n^{2/3+\varepsilon} = \tilde{O}(n^{2/3+\varepsilon}).$$

By Lemma 17, if $k \geq T$, then $\mathsf{OPT} \geqslant n^{2/3-2\varepsilon}$. Let LP be the online integral solution of LP (5) obtained by Algorithm 3. The approximation guarantee is

$$\frac{\tilde{O}(t\mathsf{LP}) + \tilde{O}(n^2/t)}{\mathsf{OPT}} \leq \frac{\tilde{O}(n^{2/3+\varepsilon}\mathsf{OPT}) + \tilde{O}(n^{4/3-\varepsilon})}{\mathsf{OPT}} = \tilde{O}(n^{2/3+\varepsilon}). \tag{6}$$

◀

3.3 Online directed Steiner forests with uniform costs

We recall that in this problem, we are given a directed graph $G = (V, E)$ and a set of terminals $D = \{(s_i, t_i) \mid i \in [k]\} \subseteq V \times V$. The goal is to find a minimum cardinality subgraph which includes an $s_i \rightsquigarrow t_i$ path for each terminal pair (s_i, t_i). We show the following theorem.

▶ **Theorem 7.** *For the online directed Steiner forest problem with uniform costs, there is a randomized polynomial time algorithm with competitive ratio $\tilde{O}(n^{2/3+\varepsilon})$ for any constant $\varepsilon \in (0, 1/3)$.*

Proof. In this problem, for each terminal pair (s_i, t_i), it suffices to have an $s_i \rightsquigarrow t_i$ path. Therefore, this problem reduces to the pairwise spanner problem by setting $d_i = \infty$ for each $i \in [k]$. The structure of the online algorithm is the same as that for online pairwise spanners with uniform lengths. We employ Algorithm 3 with a slight tweak and set $T = \lfloor n^{4/3-4\varepsilon} \rfloor$ and $t = n^{2/3+\varepsilon}$. If $k < T$, instead of adding edges of a shortest $s_i \rightsquigarrow t_i$ path, we use Theorem 1 to find an $\tilde{O}(n^{2/3+\varepsilon})$ competitive solution[3]. If $k \geq T$, then the algorithm is $\tilde{O}(n^{2/3+\varepsilon})$-competitive by (6). ◀

4 Conclusions and Open Problems

In this work, we present the first online algorithm for pairwise spanners with competitive ratio $\tilde{O}(n^{4/5})$ for general lengths and $\tilde{O}(n^{2/3+\varepsilon})$ for uniform lengths, and improve the competitive ratio for the online directed Steiner forest problem with uniform costs to $\tilde{O}(n^{2/3+\varepsilon})$ when $k = \omega(n^{4/3})$. We also show an efficient modified framework for online covering and packing. Our work raises several open questions that we state below.

An intriguing open problem is improving the competitive ratio for online pairwise spanners. For graphs with uniform edge lengths, there is a small polynomial gap between the state-of-the-art offline approximation ratio $\tilde{O}(n^{3/5+\varepsilon})$ and the online competitive ratio $\tilde{O}(n^{2/3+\varepsilon})$. For graphs with general edge lengths, we are not aware of any studies about the pairwise spanner problem. Our $\tilde{O}(n^{4/5})$-competitive online algorithm intrinsically suggests an $\tilde{O}(n^{4/5})$-approximate offline algorithm. As the approach in [34] achieves an $\tilde{O}(n/\sqrt{\mathsf{OPT}})$-approximation, we believe that the approximation ratio can be improved for the offline pairwise spanner problem, by judicious case analysis according to the cardinality of OPT.

The state-of-the-art online algorithm for Steiner forests with general costs is $\tilde{O}(k^{1/2+\varepsilon})$-competitive [28]. A natural open question is designing an $o(n)$-competitive online algorithm when k is large, and potentially extend this result to the more general buy-at-bulk network design problem. The currently best known offline approximation for Steiner forests with general costs is $O(n^{2/3+\varepsilon})$ [20], by case analysis that settles thick and thin terminal pairs separately. However, the approach in [20] for settling thin pairs is essentially a greedy procedure which is inherently offline. Our approach utilizes the uniform cost assumption to obtain a useful lower bound for the optimal solution, which is incompatible with general costs. It would be interesting to resolve the aforementioned obstacles and have an $o(n)$-competitive online algorithm for directed Steiner forests with general edge costs. One open problem for uniform costs is to improve the competitive ratio, as there is a polynomial gap between the state-of-the-art offline approximation ratio $\tilde{O}(n^{26/45+\varepsilon})$ and the online competitive ratio $\tilde{O}(n^{2/3+\varepsilon})$.

[3] One can also use the $\tilde{O}(k^{1/2+\delta})$-competitive online algorithm for graphs with general costs in [28].

References

1. Amir Abboud and Greg Bodwin. Reachability preservers: New extremal bounds and approximation algorithms. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1865–1883. SIAM, 2018.
2. Reyan Ahmed, Greg Bodwin, Faryad Darabi Sahneh, Keaton Hamm, Mohammad Javad Latifi Jebelli, Stephen Kobourov, and Richard Spence. Graph spanners: A tutorial review, 2019. `arXiv:1909.03152`.
3. Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. A general approach to online network optimization problems. *ACM Transactions on Algorithms (TALG)*, 2(4):640–660, 2006.
4. Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. The online set cover problem. *SIAM J. Comput.*, 39(2):361–370, 2009.
5. Noga Alon and Baruch Schieber. Optimal preprocessing for answering on-line product queries, 1987.
6. Spyridon Antonakopoulos. Approximating directed buy-at-bulk network design. In *International Workshop on Approximation and Online Algorithms*, pages 13–24. Springer, 2010.
7. Esther M Arkin, Joseph SB Mitchell, and Christine D Piatko. Bicriteria shortest path problems in the plane. In *Proc. 3rd Canad. Conf. Comput. Geom*, pages 153–156. Citeseer, 1991.
8. Pranjal Awasthi, Madhav Jha, Marco Molinaro, and Sofya Raskhodnikova. Testing lipschitz functions on hypergrid domains. *Algorithmica*, 74(3):1055–1081, 2016.
9. Baruch Awerbuch. Communication-time trade-offs in network synchronization. In *Proceedings of the Fourth Annual ACM Symposium on Principles of Distributed Computing*, PODC '85, page 272–276, New York, NY, USA, 1985. Association for Computing Machinery.
10. Baruch Awerbuch and Yossi Azar. Buy-at-bulk network design. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 542–547. IEEE, 1997.
11. Baruch Awerbuch, Yossi Azar, and Yair Bartal. On-line generalized steiner problem. *Theoretical Computer Science*, 324(2-3):313–324, 2004.
12. Baruch Awerbuch, Yossi Azar, and Serge Plotkin. Throughput-competitive on-line routing. In *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pages 32–40. IEEE, 1993.
13. Yossi Azar, Umang Bhaskar, Lisa Fleischer, and Debmalya Panigrahi. Online mixed packing and covering. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 85–100. SIAM, 2013.
14. Yossi Azar, Niv Buchbinder, TH Hubert Chan, Shahar Chen, Ilan Reuven Cohen, Anupam Gupta, Zhiyi Huang, Ning Kang, Viswanath Nagarajan, Joseph Naor, et al. Online algorithms for covering and packing problems with convex objectives. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 148–157. IEEE, 2016.
15. Nikhil Bansal, Niv Buchbinder, and Joseph Naor. Randomized competitive algorithms for generalized caching. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 235–244, 2008.
16. Nikhil Bansal, Niv Buchbinder, and Joseph Naor. A primal-dual randomized algorithm for weighted paging. *Journal of the ACM (JACM)*, 59(4):1–24, 2012.
17. Surender Baswana. Streaming algorithm for graph spanners - single pass and constant processing time per edge. *Inf. Process. Lett*, 2008.
18. Surender Baswana and Telikepalli Kavitha. Faster algorithms for all-pairs approximate shortest paths in undirected graphs. *SIAM J. Comput.*, 39(7):2865–2896, 2010.
19. MohammadHossein Bateni and MohammadTaghi Hajiaghayi. Euclidean prize-collecting steiner forest. *Algorithmica*, 62(3-4):906–929, 2012.
20. Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and directed steiner forest. *Information and Computation*, 222:93–107, 2013.

21 Piotr Berman and Chris Coulston. On-line algorithms for steiner tree problems. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 344–353, 1997.

22 Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P Woodruff. Transitive-closure spanners. *SIAM Journal on Computing*, 41(6):1380–1425, 2012.

23 Greg Bodwin and Virginia Vassilevska Williams. Better distance preservers and additive spanners. In Robert Krauthgamer, editor, *SODA*, pages 855–872. SIAM, 2016.

24 Glencora Borradaile, Philip N Klein, and Claire Mathieu. A polynomial-time approximation scheme for euclidean steiner forest. *ACM Transactions on Algorithms (TALG)*, 11(3):1–20, 2015.

25 Niv Buchbinder and Joseph Naor. Improved bounds for online routing and packing via a primal-dual approach. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 293–304. IEEE, 2006.

26 Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research*, 34(2):270–286, 2009.

27 Niv Buchbinder and Joseph Seffi Naor. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.

28 Deeparnab Chakrabarty, Alina Ene, Ravishankar Krishnaswamy, and Debmalya Panigrahi. Online buy-at-bulk network design. *SIAM J. Comput.*, 47(4):1505–1528, 2018.

29 Moses Charikar, Chandra Chekuri, To-yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *Journal of Algorithms*, 33(1):73–91, 1999.

30 Shuchi Chawla, Tim Roughgarden, and Mukund Sundararajan. Optimal cost-sharing mechanisms for steiner forest problems. In *International Workshop on Internet and Network Economics*, pages 112–123. Springer, 2006.

31 Shiri Chechik. Approximate distance oracles with improved bounds. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *STOC*, pages 1–10. ACM, 2015.

32 Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Transactions on Algorithms (TALG)*, 7(2):1–17, 2011.

33 Chandra Chekuri, Mohammad Taghi Hajiaghayi, Guy Kortsarz, and Mohammad R Salavatipour. Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM Journal on Computing*, 39(5):1772–1798, 2010.

34 Eden Chlamtáč, Michael Dinitz, Guy Kortsarz, and Bundit Laekhanukit. Approximating spanners and directed steiner forest: Upper and lower bounds. *ACM Transactions on Algorithms (TALG)*, 16(3):1–31, 2020.

35 Lenore Cowen and Christopher G. Wagner. Compact roundtrip routing in directed networks. *J. Algorithms*, 50(1):79–95, 2004.

36 Bilel Derbel, Cyril Gavoille, and David Peleg. Deterministic distributed construction of linear stretch spanners in polylogarithmic time. In Andrzej Pelc, editor, *DISC*, volume 4731 of *Lecture Notes in Computer Science*, pages 179–192. Springer, 2007.

37 Bilel Derbel, Cyril Gavoille, David Peleg, and Laurent Viennot. On the locality of distributed sparse spanner construction. In Rida A. Bazzi and Boaz Patt-Shamir, editors, *PODC*, pages 273–282. ACM, 2008.

38 Michael Dinitz and Robert Krauthgamer. Directed spanners via flow-based linear programs. In *STOC*, pages 323–332, 2011.

39 Michael Dinitz, Yasamin Nazari, and Zeyu Zhang. Lasserre integrality gaps for graph spanners and related problems. *arXiv preprint arXiv:1905.07468*, 2019.

40 Michael Dinitz and Zeyu Zhang. Approximating low-stretch spanners. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 821–840. SIAM, 2016.

41 Dorit Dor, Shay Halperin, and Uri Zwick. All-pairs almost shortest paths. *SIAM J. Comput.*, 29(5):1740–1759, 2000.

42 Michael Elkin. Computing almost shortest paths. *ACM Trans. Algorithms*, 1(2):283–323, 2005.

43 Michael Elkin. Streaming and fully dynamic centralized algorithms for constructing and maintaining sparse spanners. *ACM Trans. Algorithms*, 7(2):20:1–20:17, 2011.

44 Michael Elkin and David Peleg. The client-server 2-spanner problem with applications to network design. In Francesc Comellas, Josep Fàbrega, and Pierre Fraigniaud, editors, *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*, volume 8 of *Proceedings in Informatics*, pages 117–132. Carleton Scientific, 2001.

45 Michael Elkin and David Peleg. The hardness of approximating spanner problems. *Theory Comput. Syst.*, 41(4):691–729, 2007.

46 Moran Feldman, Guy Kortsarz, and Zeev Nutov. Improved approximation algorithms for directed steiner forest. *Journal of Computer and System Sciences*, 78(1):279–292, 2012.

47 Manuel Fernandez, David P. Woodruff, and Taisuke Yasuda. Graph spanners in the message-passing model. In Thomas Vidick, editor, *ITCS*, volume 151 of *LIPIcs*, pages 77:1–77:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

48 Arnold Filtser, Michael Kapralov, and Navid Nouri. Graph spanners by sketching in dynamic streams and the simultaneous communication model. *arXiv preprint arXiv:2007.14204*, 2020.

49 Lisa Fleischer, Jochen Könemann, Stefano Leonardi, and Guido Schäfer. Simple cost sharing schemes for multicommodity rent-or-buy and stochastic steiner tree. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 663–670, 2006.

50 Naveen Garg, Goran Konjevod, and Ramamoorthi Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *Journal of Algorithms*, 37(1):66–84, 2000.

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

52 Elena Grigorescu, Young-San Lin, and Kent Quanrud. Online directed spanners and steiner forests. *CoRR*, 2021. `arXiv:2103.04543`.

53 Anupam Gupta, Amit Kumar, Martin Pál, and Tim Roughgarden. 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, 2003. Proceedings.*, pages 606–615. IEEE, 2003.

54 Anupam Gupta and Viswanath Nagarajan. Approximating sparse covering integer programs online. *Mathematics of Operations Research*, 39(4):998–1011, 2014.

55 Anupam Gupta, R Ravi, Kunal Talwar, and Seeun William Umboh. Last but not least: Online spanners for buy-at-bulk. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 589–599. SIAM, 2017.

56 Anupam Gupta, Kunal Talwar, and Udi Wieder. Changing bases: Multistage optimization for matroids and matchings. In *International Colloquium on Automata, Languages, and Programming*, pages 563–575. Springer, 2014.

57 Refael Hassin. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations research*, 17(1):36–42, 1992.

58 Christopher S Helvig, Gabriel Robins, and Alexander Zelikovsky. An improved approximation scheme for the group steiner problem. *Networks: An International Journal*, 37(1):8–20, 2001.

59 Makoto Imase and Bernard M Waxman. Dynamic steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.

60 Michael Kapralov and David P. Woodruff. Spanners and sparsifiers in dynamic streams. In Magnús M. Halldórsson and Shlomi Dolev, editors, *PODC*, pages 272–281. ACM, 2014.

61 Anna R. Karlin, Mark S. Manasse, Lyle A. McGeoch, and Susan S. Owicki. Competitive randomized algorithms for non-uniform problems. *Algorithmica*, 11(6):542–571, 1994.

62 Vikram Khurana, Jian Peng, Chee Yeun Chung, Pavan K Auluck, Saranna Fanning, Daniel F Tardiff, Theresa Bartels, Martina Koeva, Stephen W Eichhorn, Hadar Benyamini, et al. Genome-scale networks link neurodegenerative disease genes to α-synuclein through specific molecular pathways. *Cell systems*, 4(2):157–170, 2017.

63 Jochen Könemann, Stefano Leonardi, Guido Schäfer, and Stefan van Zwam. 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*, pages 930–942. Springer, 2005.

64 Jochen Könemann, Stefano Leonardi, Guido Schäfer, and Stefan HM van Zwam. A group-strategyproof cost sharing mechanism for the steiner forest game. *SIAM Journal on Computing*, 37(5):1319–1341, 2008.

65 Guy Kortsarz. On the hardness of approximating spanners. *Algorithmica*, 30:432–450, 2001.

66 Dean H Lorenz and Danny Raz. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters*, 28(5):213–219, 2001.

67 Jakub Pachocki, Liam Roditty, Aaron Sidford, Roei Tov, and Virginia Vassilevska Williams. Approximating cycles in directed graphs: Fast algorithms for girth and roundtrip spanners. In Artur Czumaj, editor, *SODA*, pages 1374–1392. SIAM, 2018.

68 Mihai Patrascu and Liam Roditty. Distance oracles beyond the Thorup-Zwick bound. *SIAM J. Comput.*, 43(1):300–311, 2014.

69 David Peleg and Alejandro A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989. doi:10.1002/jgt.3190130114.

70 David Peleg and Jeffrey D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Comput.*, 18(4):740–747, 1989.

71 Leila Pirhaji, Pamela Milani, Mathias Leidl, Timothy Curran, Julian Avila-Pacheco, Clary B Clish, Forest M White, Alan Saghatelian, and Ernest Fraenkel. Revealing disease-associated pathways by network integration of untargeted metabolomics. *Nature methods*, 13(9):770–776, 2016.

72 Liam Roditty, Mikkel Thorup, and Uri Zwick. Roundtrip spanners and roundtrip routing in directed graphs. *ACM Trans. Algorithms*, 4(3):29:1–29:17, 2008.

73 Tim Roughgarden and Mukund Sundararajan. Optimal efficiency guarantees for network design mechanisms. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 469–483. Springer, 2007.

74 Xiangkun Shen and Viswanath Nagarajan. Online covering with l_q-norm objectives and applications to network design. *Mathematical Programming*, 184, 2020.

75 Andrew Chi-Chih Yao. Space-time tradeoff for answering range queries (extended abstract). In *STOC '82*, 1982.

76 Neal Young. The k-server dual and loose competitiveness for paging. *Algorithmica*, 11(6):525–541, 1994.

77 Alexander Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997.

A Additional background and related work

A model related to online algorithms is that of streaming algorithms. In the streaming model an input is also revealed sequentially, but the algorithm is only allowed to use some small amount of space, which is sublinear in the length of the stream, and is supposed to maintain an approximate solution. For this model, several papers consider spanner variants, such as undirected or weighted graphs, and additive or multiplicative stretch approximations, and the aim is to build spanners with small size or distortion [17, 48, 60]. In a related direction, spanners have also been studied in the setting of dynamic data structures, where the edges of a graph are inserted or removed one at a time and the goal is to maintain an approximate solution with small update time and space [23, 43]. A relevant model is that of distributed

computation where nodes in the network communicate efficiently to build a solution [36,37,47]. As mentioned earlier, the survey by Ahmed et al. [2] gives a comprehensive account of the vast literature on spanners, and we refer the reader to the references within.

In the buy-at-bulk network design problem [10], each edge is associated with a sub-additive cost function of its *load*. Given a set of terminal demands, the goal is to route integral flows from each source to each sink concurrently to minimize the total cost of the routing. This problem is a generalization of various single-source or multicommodity network connectivity problems, including Steiner trees and Steiner forests, in which the cost function of each edge is a fixed value once allocated. While most problems admit polylogarithmic approximations in either the online or offline setting for undirected networks [11, 21, 32, 55], the problems are much harder for directed networks. In the offline setting, the current best approximation ratio is $O(k^{\varepsilon})$ for the directed Steiner tree problem [29, 77], $O(\min\{k^{1/2+\varepsilon}, n^{2/3+\varepsilon}\})$ for the directed Steiner forest problem [20, 32], and $O(\min\{k^{1/2+\varepsilon}, n^{4/5+\varepsilon}\})$ for the directed buy-at-bulk problem [6]. In the online setting for directed networks, [28] showed that compared to offline, it suffices to pay an extra polylogarithmic factor, where the polylogarithmic term was later improved by [74]. The main contribution of [28] is essentially bringing the *junction-tree-based* approach into the online setting for connectivity problems. This is the main ingredient that improves the competitive ratio of our online algorithm from $\tilde{O}(n^{4/5+\varepsilon})$ for pairwise spanners to $\tilde{O}(n^{2/3+\varepsilon})$ for Steiner forests. Our approach for online pairwise spanners with uniform edge length combines this ingredient and the offline pairwise spanner framework [34] which tackles hard distance requirements.

As previously mentioned, generating separating constraints with an oracle in the online setting is not new. For example, this arises implicitly in early work on network optimization [3] and the oracle is discussed explicitly in [26]. As a recent example, [56] develops online algorithms for the multistage matroid maintenance problem, which requires solving a covering LP with box constraints online. [56] adjusts the separation oracle to only identify constraints that are violated by at least some constant. Because of the $\{0, 1\}$-incidence structure of their LP, the sum of primal variables has to increase by a constant to satisfy such a constraint. Meanwhile the box constraints limit the total sum of primal variables to $O(n)$. This leads to an $O(n)$ bound on the number of separating constraints. While there are strong similarities to our approach, one difference is the use of the $\{0, 1\}$-structure and box constraints to obtain their bound. Our comparably unstructured setting required us to develop an argument independent of concrete features such as these.

Beyond linear objectives, there are other variants of online covering and packing problems, which focus on different objectives with linear constraints. This includes optimizing convex objectives [14] and ℓ_q-norm objectives [74]. Other online problem-dependent variants include for instance mixed covering and packing programs [13], and sparse integer programs [54]. All these frameworks utilize the primal-dual technique, which updates the covering and packing solutions simultaneously with some judiciously selected growth rate, to guarantee nice competitive ratio. Instead, our modified framework focuses on the efficiency of online algorithms for fundamental covering and packing problems, which is amenable to applications with exponential or unbounded number of constraints, where a violating one can be searched by an efficient separation oracle.

B Proof of Theorem 2

▶ **Theorem 2.** *For the online pairwise spanner problem, there is a randomized polynomial time algorithm with competitive ratio* $\tilde{O}(n^{4/5})$.

Proof. Suppose in the online setting, there are k rounds where k may be unknown. In round $i \in [k]$, the pair (s_i, t_i) and the distance requirement d_i arrive and we select some new edges from E to settle (s_i, t_i). We run Algorithm 3 by setting $T = t = \lfloor n^{4/5} \rfloor$. It suffices to show that 1) LP (5) can be solved online by losing a polylogarithmic factor, and 2) the overall probability of edge selection is consistent with the probability based on the online solution of LP (5).

B.1 Converting and solving LP (5) online

The goal is to update x in a non-decreasing manner upon the arrival of the pair (s_i, t_i) to satisfy all its corresponding constraints, so that the objective value is still approximately optimal. We convert LP (5) into a covering LP as follows.

First, we check in round i, given the *edge capacity* x, if there is a (fractional) $s_i \rightsquigarrow t_i$ path of length at most d_i. This can be captured by checking the optimum of the following LPs,

$$\max_{y} \sum_{P \in \mathcal{P}_i} y_P \text{ over } y : \mathcal{P}_i \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{P | e \in P \in \mathcal{P}_i} y_P \leq x_e \text{ for all } e \in E \tag{7}$$

and its dual

$$\min_{z} \sum_{e \in E} x_e z_e \text{ over } z : E \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{e \in P} z_e \geq 1 \text{ for all } P \in \mathcal{P}_i. \tag{8}$$

We say that x is *good* if the optimum of LP (7) and LP (8) is at least 1, and it is *bad* otherwise. Namely, x is good if there is at least one (fractional) $s_i \rightsquigarrow t_i$ path of length at most d_i. In LP (8), the feasibility problem is equivalent to the following problem. Given the local graph G^i and edge weight z, is there an $s_i \rightsquigarrow t_i$ path of length at most d_i and weight less than 1? We note that with uniform lengths, this problem can be solved by Bellman-Ford algorithm with d_i iterations, which computes the smallest weight among all $s_i \rightsquigarrow t_i$ paths of length at most d_i in the local graph G^i.

Although this bicriteria path problem in general is NP-hard [7], an FPTAS is known to exist [57,66], which gives an approximate separation oracle. We can verify in polynomial time that if there is a path of length at most d_i and weight less than $1 - \varepsilon$. We obtain a solution z' for LP (8) where each constraint is satisfied by a factor of $1 - \varepsilon$ and set $z := z'/(1 - \varepsilon)$ as the solution.

To solve LP (5), suppose in round i, we are given x. First, we check if x is good or bad by approximately solving LP (8). If x is good, then there exists y such that $\sum_{P \in \mathcal{P}_j} y_P \geq 1$, i.e. the solution is feasible for LP (5), so we move on to the next round. Otherwise, x is bad, so we increment x until it becomes good, which implies $\sum_{e \in E} x_e z_e \geq 1$ for all feasible z in LP (8). Let Z_i be the feasible polyhedron of LP (8) in round i. We derive the following LP (essentially the one in [38,39]) which is equivalent to LP (5), by considering all the constraints of LP (8) from round 1 to round k.

$$\min_{x} \sum_{e \in E} x_e \text{ over } x : E \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{e \in E} z_e x_e \geq 1 \text{ for all } i \in [k] \text{ and } z \in Z_i. \tag{9}$$

In round $i \in [k]$, the subroutine that approximately solves LP (8) and checks if the optimum is good or not, is the separation oracle used for solving LP (9) online. Here we use Theorem 20 (the formal version of Theorem 8) to show that LP (9) can be solved online in polynomial time by paying an $O(\log n)$ factor. This requires that both $\log(1/z_e)$ and $\log \mathsf{LP}^*$ are polynomial in the number of bits used for the edge lengths, where LP^* is the

optimum of LP (9). Clearly, $\log \mathsf{LP}^* \le \log |E|$ is in $\mathsf{poly}(n)$. For $\log(1/z_e)$, the subroutine that approximately solves LP (8) returns an approximate solution z which is represented by polynomial number of bits used for the edge lengths [57, 66]. By Theorem 20, we have the following Lemma.

▶ **Lemma 19.** *There exists a polynomial time $O(\log n)$-competitive online algorithm for LP (5).*

B.2 Conditional edge selection

After having a fractional solution of LP (5) in round i where $i \ge T = \lfloor n^{4/5} \rfloor$, we independently pick $e \in E \setminus E'$ with some scaled probability so that the edge selection is consistent with the probability based on the online solution of LP (5). More specifically, let $p_e := \min\{1, x_e t \ln n\}$ and let p_e^i be the value of p_e in round i. Let $\tilde{E}$ be the set of edges where each edge is neither selected while adding cheapest feasible paths prior to round T nor selected while adding in-arborescences and out-arborescences in round T. We show that each edge $e \in \tilde{E}$ has already been selected with probability p_e^i in round i. This can be proved by induction. According to Algorithm 3, the base case is round T, where $e \in \tilde{E}$ is selected with probability p_e^T. Now suppose $i > T$, if $e \in \tilde{E}$ has been selected, it is either selected prior to round i or in round i. For the former case, e must had already been selected in round $i-1$, with probability p_e^{i-1} by inductive hypothesis. For the later case, conditioned on e has not been selected in round $i-1$, e is selected with probability $(p_e^i - p_e^{i-1})/(1-p_e^{i-1})$. Therefore, in round i, e has been selected with probability $p_e^{i-1} + (1-p_e^{i-1}) \cdot \frac{p_e^i - p_e^{i-1}}{1-p_e^{i-1}} = p_e^i$, which completes the proof. Intuitively, when $i > T$, conditioned on $e \in \tilde{E}$ was not picked from round 1 to round $i-1$, we pick e with probability $(p_e^i - p_e^{i-1})/(1-p_e^{i-1})$ at round i, so that the overall probability that e is picked from round 1 to round i is p_e^i.

B.3 Summary

We conclude the proof as follows. The overall algorithm is given in Algorithm 3. For the initialization, x is a zero vector, E' is an empty set, and $T = t = \lfloor n^{4/5} \rfloor$. The set E' is the solution. We pay an extra logarithmic factor for solving LP (5) online by Lemma 19. The competitive ratio remains $\tilde{O}(n^{4/5})$. ◀

C Online Covering in Polynomial Time

This section is devoted to proving the formal version of Theorem 8. We recall that the problem of interest is to solve the covering LP (2) online:

$$\text{minimize } \langle \mathbf{c}, x \rangle \text{ over } x \in \mathbb{R}^n_{\ge 0} \text{ s.t. } Ax \ge \mathbf{1}$$

where $A \in \mathbb{R}^{m \times n}_{\ge 0}$ consists of m covering constraints, $\mathbf{1} \in \mathbb{R}^m_{>0}$ is a vector of all ones treated as the lower bound of the covering constraints, and $\mathbf{c} \in \mathbb{R}^n_{>0}$ denotes the positive coefficients of the linear cost function.

In the online covering problem, the cost vector $\mathbf{c}$ is given offline, and each of these covering constraints is presented one by one in an online fashion, that is, m can be unknown. In round $i \in [m]$, $\{a_{ij}\}_{j \in [n]}$ (where a_{ij} denotes the i-th row j-th column entry of A) is revealed, and we have to monotonically update x so that the constraint $\sum_{j \in [n]} a_{ij} x_j \ge 1$ is satisfied. We always assume that there is at least one positive entry a_{ij} in each round i, otherwise constraint i cannot be satisfied since all the row entries are zeros. The goal is to update x in a non-decreasing manner and approximately minimize the objective value $\langle \mathbf{c}, x \rangle$.

We recall that an important idea in this line of work is to simultaneously consider the dual packing problem:

$$\text{maximize } \langle \mathbf{1}, y \rangle \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ s.t. } A^T y \leq \mathbf{c}$$

where A^T consists of n packing constraints with an upper bound $\mathbf{c}$ given offline.

The primal-dual framework in [26] simultaneously solves both LP (2) and LP (3), and crucially uses LP-duality and strong connections between the two solutions to argue that they are both nearly optimal. The modified framework closely follows the *guess-and-double* scheme in [26]. Specifically, the scheme runs in phases where each phase estimates a lower bound for the optimum. When the first constraint arrives, the scheme generates the first lower bound

$$\alpha(1) \leftarrow \min_{j \in [n]} \{ \frac{c_j}{a_{1j}} \mid a_{1j} > 0 \} \leq \mathsf{OPT}$$

where c_j is the j-th entry of $\mathbf{c}$ and OPT is the optimal value of LP (2).

During phase r, we always assume that the lower bound of the optimum is $\alpha(r)$ until the online objective $\langle \mathbf{c}, x \rangle$ exceeds $\alpha(r)$. Once the online objective exceeds $\alpha(r)$, we start the new phase $r+1$ from the current violating constraint[4] (let us call it constraint i_{r+1}, in particular, $i_1 = 1$), and double the estimated lower bound, i.e. $\alpha(r+1) \leftarrow 2\alpha(r)$. We recall that x must be updated in a non-decreasing manner, so the algorithm maintains $\{x_j^r\}$, which denotes the value of each variable x_j in each phase r, and the value of each variable x_j is actually set to $\max_r \{x_j^r\}$.

In Algorithm 4, we describe one round of the modified scheme in phase r. When a covering constraint i arrives, we introduce a packing variable $y_i = 0$. If the constraint is violated, we increment each x_j according to an exponential function of y_i until the constraint is satisfied *by a factor of 2*. This is the main difference between the modified framework and [26], which increments the variables until the constraint is satisfied.

Algorithm 4 Online Covering.

for arriving covering constraint i **do**
  $y_i \leftarrow 0$. ▷ the packing variable y_i is used for the analysis
  if $\sum_{j=1}^n a_{ij} x_j^r < 1$ **then** ▷ if constraint i is not satisfied
    while $\sum_{j=1}^n a_{ij} x_j^r < 2$ **do** ▷ update until constraint i is satisfied by a factor of 2
      Increase y_i continuously.
      Increase each variable x_j^r by the following increment function:

$$x_j^r \leftarrow \frac{\alpha(r)}{2nc_j} \exp\left(\frac{\ln(2n)}{c_j} \sum_{k=i_r}^{i} a_{kj} y_k \right).$$

Although the augmentation is in a continuous fashion, it is not hard to implement it in a discrete way for any desired precision by binary search. Therefore, to show that the modified framework is efficient, it suffices to bound the number of violating constraints it will encounter. The performance of the modified scheme is analyzed in Theorem 20 (the formal version of Theorem 8).

[4] In [26], the scheme starts all over again from the first constraint. We start from the current violating constraint because it is more amenable when violating constraints are generated by a separation oracle. There is no guarantee for the order of arriving violating constraints in such settings.

▶ **Theorem 20.** *There exists an $O(\log n)$-competitive online algorithm for the covering LP (2) which encounters $\mathsf{poly}(n, \log \mathit{OPT}, \log(1/\alpha(1)))$ violating constraints.*

Proof. The proof for the $O(\log n)$-competitiveness closely follows the one in [26]. Let $X(r)$ and $Y(r)$ be the covering and packing objective values, respectively, generated during phase r. The following claims are used to show that Algorithm 4 is $O(\log n)$-competitive.

1. x is feasible.
2. For each *finished* phase r, $\alpha(r) \le 4\ln(2n) \cdot Y(r)$.
3. y generated during phase r is feasible.
4. The sum of the covering objective generated from phase 1 to r is at most $2\alpha(r)$.
5. Let r' be the last phase, then the covering objective $\langle \mathbf{c}, x\rangle \le 2\alpha(r')$.

From these five claims together with weak duality, we conclude that

$$\langle \mathbf{c}, x\rangle \le 2\alpha(r') = 4\alpha(r'-1) \le 16\ln(2n) \cdot Y(r'-1) \le 16\ln(2n) \cdot \mathsf{OPT}.$$

Now we show that Algorithm 4 encounters $\mathsf{poly}(n, \log \mathsf{OPT}, \log(1/\alpha(1)))$ violating constraints. We first show that there are $O(\log\log n + \log \mathsf{OPT} + \log(1/\alpha(1)))$ phases. The estimated lower bound α doubles when we start a new phase. Suppose there are r' phases, then $\alpha(1) \cdot 2^{r'-1} = O(\log n)\mathsf{OPT}$ because Algorithm 4 is $O(\log n)$-competitive. This implies that $r' = O(\log\log n + \log \mathsf{OPT} + \log(1/\alpha(1)))$.

In each phase, when a violating constraint arrives, we increment x so that the constraint is satisfied by a factor of 2. This implies that at least one variable x_j is doubled. $x_j = O(\log n)\mathsf{OPT}/c_j$ because $c_j x_j \le \langle \mathbf{c}, x\rangle = O(\log n)\mathsf{OPT}$. At the start of phase r, $x_j = \alpha(r)/(2nc_j) \ge \alpha(1)/(2nc_j)$. Suppose x_j has been doubled t times in phase r, then

$$\frac{\alpha(1)}{2nc_j} \cdot 2^t \le \frac{\alpha(r)}{2nc_j} \cdot 2^t \le x_j = O(\log n)\frac{\mathsf{OPT}}{c_j}$$

which indicates that $t = O(\log n + \log \mathsf{OPT} + \log(1/\alpha(1)))$.

There are n variables and r' phases, and in each phase, each variable is doubled at most t times. Therefore, Algorithm 4 encounters $\mathsf{poly}(n, \log \mathsf{OPT}, \log(1/\alpha(1)))$ violating constraints. ◀