Approximating Spanners and Directed Steiner Forest: Upper and Lower Bounds

Eden Chlamtáč[*]
Ben Gurion University

Michael Dinitz
Johns Hopkins University

Guy Kortsarz
Rutgers University-Camden

Bundit Laekhanukit
Weizmann Institute of Science

August 3, 2016

Abstract

It was recently found that there are very close connections between the existence of *additive spanners* (subgraphs where all distances are preserved up to an additive stretch), *distance preservers* (subgraphs in which demand pairs have their distance preserved exactly), and *pairwise spanners* (subgraphs in which demand pairs have their distance preserved up to a multiplicative or additive stretch) [Abboud-Godwin SODA '16, Godwin-Williams SODA '16]. We study these problems from an optimization point of view, where rather than studying the existence of extremal instances we are given an instance and are asked to find the sparsest possible spanner/preserver. We give an $O(n^{3/5+\varepsilon})$-approximation for distance preservers and pairwise spanners (for arbitrary constant $\varepsilon > 0$). This is the first nontrivial upper bound for either problem, both of which are known to be as hard to approximate as Label Cover. We also prove Label Cover hardness for approximating additive spanners, even for the cases of additive 1 stretch (where one might expect a polylogarithmic approximation, since the related multiplicative 2-spanner problem admits an $O(\log n)$-approximation) and additive polylogarithmic stretch (where the related multiplicative spanner problem has an $O(1)$-approximation)

Interestingly, the techniques we use in our approximation algorithm extend beyond distance-based problem to pure connectivity network design problems. In particular, our techniques allow us to give an $O(n^{3/5+\varepsilon})$-approximation for the Directed Steiner Forest problem (for arbitrary constant $\varepsilon > 0$) when all edges have uniform costs, improving the previous best $O(n^{2/3+\varepsilon})$-approximation due to Berman et al. [ICALP '11] (which holds for general edge costs).

---

1   Introduction

There has been significant recent progress on problems involving *graph spanners*: subgraphs which approximately preserve distances. The traditional notion of spanner has involved multiplicative stretch, in which all distances are preserved up to a multiplicative factor known as the *stretch*. If this stretch factor is k then the subgraph is known as a *k-spanner*. It has been known for over 20 years that all undirected graphs admit sparse spanners if the stretch factor is at least 3, where sparser and sparser spanners are possible as the stretch factor increases. More formally, Althöfer et al. [4] showed that for every graph G and integer $k \geq 1$, there is a $(2k-1)$-spanner of G with at most $n^{1+1/k}$ edges. Moreover, this is tight assuming the Erdős girth conjecture [22].

A different but synergistic question involves *optimizing* spanners: given an input graph G and a stretch value k, can we algorithmically find the *sparsest k-spanner* of G? After all, not all graphs are those from the girth conjectures, and if some graph does have a sparse spanner, we would like to find it. This is known as the BASIC k-SPANNER problem [17], and similar optimization problems can be defined for many other versions of spanners where tradeoffs do not exist, such as for directed graphs [8, 18, 7, 20], when the objective is to minimize the maximum degree rather than the sparsity [14, 15], etc. Many of these problems can be thought of as standard network design problems (e.g. Steiner Tree, Steiner Forest, etc.) but where the connectivity constraint is augmented with a distance constraint: not only do certain nodes need to be connected, the distance of the connecting path must be short. There has been significant recent progress, both positive and negative, on many of these problems: in general they tend to be as hard to approximate as Label Cover [5] (but not always), but it is still often possible to give nontrivial approximation algorithms.

In parallel with this work on optimizing spanners, there has been rapid progress on understanding what other tradeoffs are possible. Three directions in particular have been the focus of much of this work, and in fact have been shown to be related [2, 9]: *additive spanners*, *distance preservers*, and *pairwise spanners*. In additive spanners we restrict attention to unweighted graphs, but ask for the stretch to be additive rather than multiplicative (giving us more leeway for small distances, but less flexibility for long distances). In distance preservers and pairwise spanners we make an orthogonal change to the spanner definition: instead of preserving *all* distances, we are given some subset of pairs of nodes $T \subseteq V \times V$ (known as the demands) and are only required to preserve distances between demand pairs. In a preserver we must preserve these distances *exactly*, while in a pairwise spanner we again allow some stretch (multiplicative or additive) for the demands. All three of these objects exhibit somewhat surprising behavior (see Section 1.2 for a more detailed discussion), but have also shown to be related to each other (for example, preserver lower bounds can imply pairwise spanner lower bounds [2], and some overlapping techniques have been useful for both preservers and additive spanners [9]).

In this paper we bring together these lines of work, by studying the approximability of distance preservers, pairwise spanners, and additive spanners. We provide the first nontrivial upper bounds for distance preservers and pairwise spanners, while for additive spanners we provide the first known hardness of approximation. Our hardness of approximation in some ways mirrors recent results on tradeoffs for additive spanners [1], in that we can provide hardness for settings where the associated multiplicative problem is actually easy. Moreover, the techniques required for our upper bounds also yield new insight into classical network design problems, giving an improved approximation for the Directed Steiner Forest problem when all edges have the same cost.

1.1 Our Results and Techniques

We now give our results, and for each result some short intuition about the techniques used. Given a graph G, we will let $d_G(u, v)$ be the shortest-path distance between u and v in G (note that we are assuming unweighted graphs throughout this paper, so the distance is the same as the number of edges). If G is clear from context we may omit it, and simply refer to $d(u, v)$.

Definition 1.1 *Given a directed graph $G = (V, E)$ and a collection $P \subseteq V \times V$, a subgraph $H = (V, E')$ is a* pairwise distance preserver *if $d_H(u, v) = d_G(u, v)$ for all $(u, v) \in P$.*

Definition 1.2 *In the* PAIRWISE DISTANCE PRESERVER *problem we are given a graph $G = (V, E)$ (possibly directed and with edge lengths) and a collection $P \subseteq V \times V$, and are asked to return a pairwise distance preserver H which minimizes $|E(H)|$.*

Our first result is for distance preservers, where we give the first nontrivial upper bound.

Theorem 1.3 *For any constant $\epsilon > 0$, there is a polynomial-time $O(n^{3/5+\epsilon})$-approximation algorithm for the* PAIRWISE DISTANCE PRESERVER *problem.*

The main difficulty in proving this theorem is the fact that n is not a lower bound on the optimal solution. Most approximation algorithms for spanners (e.g. [8, 18, 7, 20]) involve randomly sampling shortest-path trees: since $n - 1$ is a lower bound on OPT for spanners, sampling f trees only costs us f in the approximation ratio. But for preservers, n is no longer a lower bound since the optimal solution need not be connected. Hence sampling shortest path trees is no longer low cost, and without this step any existing algorithm for spanners we might apply has unbounded cost.

To overcome this, we replace shortest-path trees with *junction trees*. Junction trees are trees which cover "significant" demand at "little" cost. Typically they involve a root node r, and a collection of shortest paths into r and shortest paths out of r in order to satisfy some of the demand. Due to their simple structure we can usually find the densest junction tree (the tree with best ratio of demand covered to cost incurred) relatively efficiently, and then due to their high density these junction trees can be combined into a global solution at low cost. They have been used extensively in network design since their introduction by [12], but this is the first time (as far as we are aware) that they have been used for spanner problems, or for any problem with a hard distance constraint. We believe that bringing this technique into spanners is a significant contribution of this work.

By using some of the ideas about junction trees that we developed for the PAIRWISE DISTANCE PRESERVER problem, we can give an improved approximation algorithm for a classic network design problem: DIRECTED STEINER FOREST with uniform costs.

Definition 1.4 *In the* DIRECTED STEINER FOREST *problem we are given a directed graph $G = (V, E)$, nonnegative edge costs $c : E \to \mathbb{R}^{\geq 0}$, and a collection of node pairs $P \subseteq V \times V$. If all edges have the same cost, we say that the edge costs are* uniform. *We are asked to return a subgraph H of G which minimizes $\sum_{e \in E(H)} c(e)$ subject to there being a directed path from s to t for all $(s, t) \in P$.*

Theorem 1.5 *For any constant $\epsilon > 0$, there is a polynomial-time $O(n^{3/5+\epsilon})$-approximation algorithm for the* DIRECTED STEINER FOREST *problem with uniform edge costs.*

The current best approximation for DIRECTED STEINER FOREST is $\tilde{O}(n^{2/3})$ [7], and also comes from intuition about spanners (directed spanners rather than distance preservers). Note that while our approximation is stronger, it is for a significantly simplified setting (uniform edge costs).

Our next result is for pairwise spanners, where we relax the exact distance requirement of preservers but unlike in DIRECTED STEINER FOREST do still have a hard restriction on the length.

Definition 1.6 *Given a directed graph $G = (V, E)$, a collection $P \subseteq V \times V$, and an integer k, a subgraph H of G is a* pairwise spanner *with multiplicative stretch k if $d_H(u, v) \leq k \cdot d_G(u, v)$ for all $(u, v) \in P$. If $d_H(u, v) \leq d_G(u, v) + k$ for all $(u, v) \in P$ then we say that H has additive stretch k.*

Our approximation algorithm will actually be for a much more general problem than finding pairwise spanners with a given stretch bound: we will allow every demand pair to have *its own* stretch bound. We can instantiate these bounds to give multiplicative or additive stretch, but we can also be more flexible if desired. This makes the problem essentially DIRECTED STEINER FOREST but with a distance bound for each demand.

Definition 1.7 *In the* PAIRWISE SPANNER *problem we are given a graph $G = (V, E)$ (possibly directed), a collection $P \subseteq V \times V$, and a function $D : P \to \mathbb{B}$. We are asked to return a subgraph H minimizing $|E(H)|$ subject to $d_H(u, v) \leq D(u, v)$ for all $(u, v) \in P$.*

Theorem 1.8 *For any constant $\epsilon > 0$, there is a polynomial-time $O(n^{3/5+\epsilon})$-approximation algorithm for the* PAIRWISE SPANNER *problem.*

As before, we would like to find a junction tree with a common root r through which we connect many terminal pairs. However, because of the distance constraints, we cannot use the [13] algorithm for junction trees as a black box (and the thresholding argument for DIRECTED STEINER FOREST does not apply due to the hard distance constraints). If we had constraints on the distances between each terminal and *the root r*, then this could be handled using a reduction which layers the graph. However, the stretch bound for a terminal pair here translates to a bound on the *sum* of distances of the two terminals in that pair to/from r. The need to coordinate these distance pairs across many terminal pairs, when the LP of [13] might spread its weight across many possible distances for each terminal, makes it much more difficult to round this LP. So we need to go into the details of [13] and significantly change the LP and the rounding in order to allow terminals to make individual distance choices but in a way that their sums still satisfy the stretch requirements. This involves major technical challenges and forms one of the main technical contributions of this paper.

Finally, we move to lower bounds by studying additive spanners.

Definition 1.9 *A subgraph H of a (possibly directed) graph $G = (V, E)$ is a $+k$-spanner if $d_H(u, v) \leq d_G(u, v) + k$ for all $u, v \in V$.*

Definition 1.10 *In the* ADDITIVE k-SPANNER *problem we are given a (possibly directed) graph $G = (V, E)$ and an integer $k > 0$, and are asked to return a $+k$-spanner H minimizing $|E(H)|$.*

To the best of our knowledge, ADDITIVE k-SPANNER has never been considered before this paper. But while not explicit in their paper, it is straightforward to see that the $\tilde{O}(n^{1/2})$-approximation algorithm of [7] for the multiplicative version continues to hold in the additive setting.

Theorem 1.11 *For any constant $\epsilon > 0$ and any value $k \geq 1$ (not necessarily constant), there is no polynomial-time $2^{\log^{1-\epsilon} n}/k^3$-approximation algorithm for* ADDITIVE k-SPANNER *unless $NP \subseteq DTIME(2^{\mathrm{polylog}(n)})$.*

Note that this theorem implies strong hardness results for both small and large extremes of k: the problem is hard when $k = 1$, but is also hard when k is polylogarithmic in n. The associated multiplicative-stretch problem, of multiplicative 2 and polylogarithmic in n respectively, are actually both easy: there is an $O(\log n)$-approximation for the multiplicative 2 problem [26], and the classical tradeoff of [4] implies an $O(1)$-approximation when the multiplicative stretch is at least $\Omega(\log n)$. The $+1$-spanner hardness is particularly surprising, as many of the techniques that have been used to prove hardness for multiplicative spanners in the past [14, 21, 27] break down for this problem. In particular, essentially all previous hardness results for spanners have been reductions from LABEL COVER [5] or MIN-REP [27] in a way where the key factor to the hardness is the difficulty of spanning some set of crucial *edges* using paths of length at most the allowed stretch. This length must be at least 3, since multiplicative 2-spanners are easier to approximate [26]. But for $+1$-spanners it cannot be that edges are what are hard to span, for the same reason (multiplicative 2-spanners are easy). Instead it must be pairs at longer distance which are hard to span. But the obvious ways of turning an edge into a longer path (e.g. subdividing) run into something of a catch-22, since spanning the intermediate edges of such a path is itself expensive. So we are forced to use a much more complicated reduction which "unifies" these paths in such a way that intermediate edges are no longer too costly, making our reduction significantly more complex.

Once we have these ideas, we can further modify them to allow us to prove hardness for much larger stretch values. Without using the ideas from the $+1$ case it is easy to prove a theorem similar to the hardness of BASIC k-SPANNER proved in [17] of $2^{(\log^{1-\epsilon} n)/k}$. This involves starting with an instance of MIN-REP and subsampling the superedges to get a high girth instance, but with this technique it is not possible to prove hardness for stretch values that are logarithmic or larger. But if we replace the superedges with longer paths, as we did in the $+1$ case, we can move the k from the exponent to a multiplicative factor, allowing hardness for much larger stretch values. This requires even more technical work than in the $+1$-case, as the longer paths introduce even more troublesome complications.

1.2 Related Work

Distance preservers were introduced by Coppersmith and Elkin [16], who showed that extremely sparse preservers exist if the number of demand pairs is not too large. The state of the art is due to Bodwin and Williams [9], who gave a more fine-grained existential analysis and also (with [2]) demonstrated some connections to additive spanners. There has been much more work on additive spanners (see [11] for a recent survey); some upper bound highlights include a $+2$-spanner with $O(n^{3/2})$ edges [3], a $+4$-spanner with $\tilde{O}(n^{7/5})$ edges [10], and $+6$-spanner with $O(n^{4/3})$ edges [6, 29]. From a lower bound perspective, Woodruff [28] gave the first lower bounds that did not depend on the Erdős girth conjecture, and recently Abboud and Bodwin [1] showed that the 4/3 exponent is tight: sparser spanners are not always possible even with the allowed additive stretch is a small polynomial.

Work on optimizing spanners began over 20 years ago, with the first upper bounds provided by Kortsarz and Peleg [26] for stretch 2 and the first lower bounds given by Kortsarz [27] and by Elkin and Peleg [21] (who in particular showed hardness for pairwise spanners; hardness for distance preservers is folklore). There has recently been a spurt of progress on these problems, including new algorithms for low stretch [7, 20], for directed spanners [8, 7, 18], for low-degree spanners [14, 15], and for fault-tolerant spanners [18, 19, 20], as well as lower bounds which finally provide strong hardness [17].

2 An $n^{3/5+\varepsilon}$-approximation for Pairwise Distance Preserver

In this section we prove Theorem 1.3. We begin with some definitions and previous results which will help. For any pair $(s,t) \in P$, let $\mathcal{P}_{s,t}$ denote the collection of shortest paths from s to t, and let the *local graph* $G^{s,t} = (V^{s,t}, E^{s,t})$ be the union of all nodes and edges in $\mathcal{P}_{s,t}$. Note that every path from s to t in $G^{s,t}$ is a shortest path. For any k, we say a pair $(s,t) \in P$ is *k-thick* if $|V^{s,t}| \geq k$ and otherwise, we call the pair *k-thin*. We say that an edge set $F \subseteq E$ *satisfies* a pair $(s,t) \in P$ is F contains a shortest path from s to t. Consider the following standard LP relaxation (essentially that of [18] for spanners).

$$
\begin{aligned}
\min \quad & \textstyle\sum_{e \in E} x_e \\
\text{s.t.} \quad & \textstyle\sum_{p \in \mathcal{P}_{s,t}} f_p \geq 1 & \forall (s,t) \in P \\
& \textstyle\sum_{p \in \mathcal{P}_{s,t}: e \in P} f_p \leq x_e & \forall (s,t) \in P, \forall e \in E^{s,t} \\
& x_e \geq 0 & \forall e \in E
\end{aligned}
$$

Let LP denote the value of the optimal solution to this relaxation. The next claim is due to [7] (here we adapt the wording for preservers).

Claim 2.1 ([7]) *Independent randomized rounding where we retain each edge e independently with probability $\min\{1, x_e \cdot k \ln n\}$ preserves the distance for every k-thin pair with high probability, and the number of edges retained is at most $\tilde{O}(k) \cdot \mathrm{LP}$.*

For thick pairs, we have a different guarantee. As noted in [8, 18, 7], the local graphs of thick pairs have a small hitting set (this can be proved by random selection or through rounding a feasible solution for a Hitting Set LP):

Claim 2.2 *We can find in polynomial a hitting set of size $\tilde{O}(n/k)$ for the vertex sets $V^{s,t}$ of all k-thick pairs (s,t).*

We assume that we know the value OPT (this is without loss of generality since we can just try every possible value in $[|E|]$ for OPT). We start with the following algorithm.

Algorithm 1: Set $k = n/\sqrt{\mathsf{OPT}}$. Run independent randomized rounding as in Claim 2.1 to get a set of edges F. Find a hitting set $X \subseteq V$ of cardinality $\tilde{O}(n/k)$ using Claim 2.2 for the vertex sets $V^{s,t}$ for k-thick pairs (s,t), and for every $u \in X$, add to F a shortest-path tree from u and a shortest-path tree into u. Return F.

It is easy to see that the resulting set is a preserver, and that the cardinality of the set of edges added to F in the final step is $\tilde{O}(n^2/k)$, and so we get the following guarantee:

Lemma 2.3 *Algorithm 1 gives an $\tilde{O}(n/\sqrt{\mathsf{OPT}})$-approximation for* Pairwise Distance Preserver.

Proof: By the above observation for thick pairs, and Claim 2.1 for thin pairs, we get an approximation guarantee of $\frac{\tilde{O}(k \cdot \mathrm{LP}) + \tilde{O}(n^2/k)}{\mathsf{OPT}} \leq \frac{\tilde{O}(k \cdot \mathsf{OPT}) + \tilde{O}(n^2/k)}{\mathsf{OPT}} = \frac{\tilde{O}(n\sqrt{\mathsf{OPT}})}{\mathsf{OPT}} = \tilde{O}\left(\frac{n}{\sqrt{\mathsf{OPT}}}\right).$ ∎

Remark 2.4 *For global spanner problems (when $\mathsf{OPT} = \Omega(n)$), a similar algorithm and analysis in [7] gives an approximation ratio of $\tilde{O}(n/\sqrt{\mathsf{OPT}}) = \tilde{O}(\sqrt{n})$. However, since we have no lower bound on OPT for non-global spanning and connectivity problems, we cannot achieve such a guarantee using just this algorithm.*

When $\mathsf{OPT} \geq n^{4/5}$, the above guarantee immediately implies our desired $\tilde{O}(n^{3/5})$-approximation. So let us now consider the case when $\mathsf{OPT} < n^{4/5}$. We begin by partitioning our pairs P into $\log n$ buckets, such that in each bucket, the distances $d(s,t)$ in that bucket are all in the range $[d^*, 2d^*)$ for some value d^* associated with that bucket. We run our algorithm separately for each bucket, at the cost of an additional $\log n$ factor in our approximation guarantee (relative to the approximation that we will state). Let us focus on one such bucket P_{d^*} for a fixed value d^*. Our next algorithm is similar to Algorithm 1, but does not add a full shortest-path tree and works well for buckets where d^* is small. For the sake of this algorithm, let us define sets of nearby terminals for every vertex $u \in V$: $S_{d^*}^u = \{s \mid d(s,u) < 2d^*, \exists t : (s,t) \in P_{d^*}\}$, and $T_{d^*}^u = \{t \mid d(u,t) < 2d^*, \exists s : (s,t) \in P_{d^*}\}$

Our second algorithm is as follows:

Algorithm 2: Set $k = \sqrt{d^*n}$. Run independent randomized rounding as in Claim 2.1 to get an edge set F. Find a hitting set $X \subseteq V$ of cardinality $\tilde{O}(n/k)$ for the vertex sets $V^{s,t}$ for k-thick pairs $(s,t) \in P_{d^*}$, and for every $u \in X$, add to F a shortest path from u to every $t \in T_{d^*}^u$ and from every $s \in S_{d^*}^u$ to u.

It is easy to see that the resulting set preserves distances for all pairs in P_{d^*}, and that the cardinality of the set of edges added to F in the final step is $\tilde{O}(d^*(\max_{u \in V}(|S_{d^*}^u| + |T_{d^*}^u|))n/k)$, and so we get the following guarantee:

Lemma 2.5 *Algorithm 2 satisfies all pairs in P_{d^*} and has at most $\tilde{O}(\sqrt{d^*n} \cdot \mathsf{OPT})$ edges.*

Proof: Note that every terminal must have at least one incident edge in any preserver, and so the number of terminals is always at most $O(\mathsf{OPT})$. Thus, by the above observation for thick pairs, and Claim 2.1 for thin pairs, we get a total cost of $\tilde{O}(k \cdot \mathsf{LP}) + \tilde{O}(d^*(\max_{u \in V}(|S_{d^*}^u| + |T_{d^*}^u|))n/k) \leq \tilde{O}(k \cdot \mathsf{OPT}) + \tilde{O}(d^*\mathsf{OPT}n/k) = \tilde{O}(\sqrt{d^*n} \cdot \mathsf{OPT})$. ∎

Note that for $d^* \leq n^{1/5}$, this gives us our desired $\tilde{O}(n^{3/5})$ approximation guarantee. Finally, we focus on the remaining case when $d^* > n^{3/5}$ (and recall that $\mathsf{OPT} < n^{4/5}$). For this regime, we use the junction tree approach which has been used until now only for problems without hard distance constraints [13, 23, 7]. A junction tree is a disjoint[1] union of an in-arborescence and an out-arborescence with terminal leaves and a common root. In our case, the leaves of the in-arborescence will come from the set $S_{d^*} = \{s \mid \exists t : (s,t) \in P_{d^*}\}$ and the leaves of the out-arborescence will come from the set $T_{d^*} = \{t \mid \exists s : (s,t) \in P_{d^*}\}$. The *density* of a junction tree (in the context of preservers) is the ratio between the number of edges in the junction tree, and the number of terminal pairs in P_{d^*} such that the junction tree contains a shortest path connecting that pair. In order to find such a junction tree, we construct a pair of graphs for every node u: we let $G_{d^*}^{\mathrm{in}}(u)$ be the union of all shortest paths into u, let $G_{d^*}^{\mathrm{out}}(u)$ be the union of all shortest paths coming out of u. Note that *every* path in these two graphs is a shortest path.

Chekuri et al. [13] show that for any fixed u, a least dense junction tree for general connectivity problems can be approximated within an n^ε factor for any $\varepsilon > 0$. While their algorithm is only designed to handle connectivity demands, and not distance-based demands, we can apply their algorithm as a black box on the graph G_{d^*}, which we define as the disjoint union of $G_{d^*}^{\mathrm{in}}(u)$ and $G_{d^*}^{\mathrm{out}}(u)$ (connected through u), and the pairs $P_{d^*}(u) = \{(s,t) \in P_{d^*} \mid d(s,u) + d(u,t) = d(s,t)\}$. Since the *only* paths connecting such terminal pairs in this graph are shortest paths, we are guaranteed that every pair connected by the algorithm of [13] will also be distance-preserved. Thus our third algorithm is as follows:

---

[1]The two arborescences may overlap in the original graph, but we may think of them as coming from two distinct copies of the graph.

Algorithm 3: As long as $P_{d^*} \neq \emptyset$, repeat the following:

- For every $u \in V$, use the algorithm of [13] to find an approximately optimal junction tree in graph $G_{d^*}(u)$ and pairs $P_{d^*}(u)$, and let F^u and P^u be the set of edges used and terminals connected by this algorithm, respectively.

- Choose u^* minimizing the ratio $|F^{u^*}|/|P^{u^*}|$, add edge set F^{u^*} to F, and remove pairs P^{u^*} from P_{d^*}.

The approximation guarantee of this algorithm is as follows:

Lemma 2.6 *Algorithm 3 satisfies all pairs in P_{d^*} and adds at most $n^\varepsilon \cdot \mathsf{OPT}^2/d^*$ edges.*

Proof: As shown in [23], the above algorithm adds at most $O(\alpha \cdot \mathsf{OPT})$ edges (for some $\alpha > 1$) as long as we have the guarantee that we can find a junction tree with density at most $\alpha \cdot \mathsf{OPT}/|P_{d^*}|$.[2] Thus, by our adaptation of the algorithm of [13], it suffices to show that there exists a junction tree with density at most $O((\mathsf{OPT}/d^*) \cdot \mathsf{OPT}/|P_{d^*}|)$, guaranteeing that we can find a junction tree with at most the same density up to an additional n^ε factor. To see that such a junction tree exists, let $F \subseteq E$ be some optimum solution, for every pair $(s,t) \in P_{d^*}$ fix a shortest path $p_{s,t}$ in F, and for every edge $e \in F$, let $P_{d^*}(e) = \{(s,t) \in P_{d^*} \mid e \in p_{s,t}\}$. Thus, for the average edge in F we have $\frac{1}{\mathsf{OPT}} \sum_{e \in F} |P_{d^*}(e)| = \frac{1}{\mathsf{OPT}} \sum_{(s,t) \in P_{d^*}} |p_{s,t}| \geq \frac{1}{\mathsf{OPT}} \sum_{(s,t) \in P_{d^*}} 2d^* = \frac{2d^*|P_{d^*}|}{\mathsf{OPT}}$. Thus there exists an edge e with a junction tree going through it which connects at least $\Omega(d^*|P_{d^*}|/\mathsf{OPT})$ pairs, and uses at most OPT edges (since it is a subset of F), giving a density of at most $O(\mathsf{OPT}/(d^*|P_{d^*}|/\mathsf{OPT})) = O(\mathsf{OPT}^2/(d^*|P_{d^*}|))$, as required. ∎

So by applying Algorithm 2 for buckets where $d^* \leq n^{1/5}$ and Algorithm 3 for buckets where $d^* > n^{1/5}$, and using the fact that $\mathsf{OPT} < n^{4/5}$ (or else Algorithm 1 is already sufficient), Lemmas 2.5 and 2.6 imply that we get an approximation ratio of at most $\tilde{O}(\sqrt{n^{1/5}n}) + O(n^\varepsilon n^{4/5}/n^{1/5}) = O(n^{3/5+\varepsilon})$, as claimed.

3   Directed Steiner Forest with Uniform Costs

We now turn to DIRECTED STEINER FOREST with uniform cost edges, but give only a brief overview; details are in Appendix A. For this problem, we would like to adapt our algorithm for preservers and get an $n^{3/5+\varepsilon}$-approximation. This could be done quite directly, if we knew, for some optimum solution $F \subseteq E$, what the distance in F of every terminal pair is. However, since these distances are not known (unlike for preservers), we need to be slightly more careful. We can use the following standard LP relaxation:

$$
\begin{aligned}
\min \quad & \sum_{e \in E} x_e \\
\text{s.t.} \quad & \text{capacities } x_e \text{ support a one unit } s-t \text{ flow } f_{s,t} && \forall (s,t) \in P \\
& x_e \geq 0 && \forall e \in E
\end{aligned}
$$

While we have no notion of local graphs independent of the LP, for every pair $(s,t) \in P$, we can define $V^{s,t}$ to be the set of vertices involved in the flow $f_{s,t}$, and we can define k-thick and k-thin

---

[2]They showed this in the context of Directed Steiner Forest, however the same proof applies for any terminal-pair demand problem.

terminal pairs w.r.t. the cardinality of this set. It follows from the analysis in [7] that k-thin pairs can be connected with cost $\tilde{O}(k \cdot \mathrm{LP})$.[3]

The algorithm proceeds along the similar lines as before: We guess OPT. If $\mathsf{OPT} \geq n^{4/5}$, we run Algorithm 1 (with the new definition of $V^{s,t}$), and get an $\tilde{O}(n^{3/5})$-approximation as before (as in Lemma 2.3). Otherwise, we repeatedly find an edge set with minimum density (relative to the number of newly connected terminal pairs). We set $D = n^{1/5}$, and try two options to find a sparse set (corresponding to two cases), only using the sparser of the two sets. The cases are as follows: If at least half the remaining unconnected pairs have distance at least D in F, we can run Algorithm 3 just as before, with the same density guarantee as in the proof of Lemma 2.6. On the other hand, if at least half the remaining unconnected pairs P' have distance at most D in F, we can write an LP relaxation in which the total flow along paths of length at most D is at least $|P'|/2$. On this relaxation, we run Algorithm 2, with threshold $k = \sqrt{Dn}$ and bound D on the distance of shortest paths to/from the hitting set. The analysis is very similar to that of Lemma 2.5.

4 Pairwise spanners

Much of our algorithm for pairwise preservers can be used for pairwise spanners with almost no change. In particular, Algorithms 1 and 2 work just as stated, with the same guarantees, up to a small (but well known) change in the LP to allow paths within the stretch bound for each pair rather than just shortest paths. However, Algorithm 3 does not work as stated. Recall that in Algorithm 3, we constructed a graph of shortest paths through a root u for which we wanted to find a sparsest junction tree using the algorithm of [13] as a black box. This black box reduction worked because we restricted our instance so that all paths were shortest paths. However, there is no obvious graph we can restrict to for arbitrary distance bounds (or even when we only allow additive or multiplicative stretch). Consider a single pair (s, t), and all paths of distance $D(s, t)$ that go through a proposed junction tree root u. Some of these paths could have a prefix of length $D(s, t)/2$ to u and a suffix of length $D(s, t)/2$ from u to t, while others may have a prefix of length $D(s, t)/2 - 1$ to u and a suffix of length $D(s, t)/2 + 1$ from u to t. Allowing the algorithm to connect s and t using the edges from the union of such paths can result in a path of length $D(s, t) + 1$, or even more, violating the required distance.

Thus handling these hard distance requirements introduces significant difficulties, and requires us to modify the algorithm of [13] in a non-black-box manner. At a high level, their algorithm uses height reduction and a reduction to (undirected) Group Steiner Tree via a density LP. Roughly speaking, our version of their density LP has a flow f_s to u for every left terminal s, and a flow f_t from u to t for every right terminal t. Each flow is explicitly decomposed into flows on specific path lengths. That is, f_s^k only uses paths of length exactly k, etc. Our LP is roughly of the form

$$
\begin{aligned}
\min \quad & \sum_{e \in E} x_e \\
\text{s.t.} \quad & \text{capacities } x_e \text{ support } s - u, u - t \text{ flows } f_s, f_t && \forall \text{ terminals } s, t && (4.1) \\
& \sum_{(s,t) \in P} \sum_{k + \ell \leq D(s,t)} \min\{|f_s^k|, |f_t^\ell|\} = 1 && && (4.2) \\
& x_e \geq 0 && \forall e \in E && (4.3)
\end{aligned}
$$

In order to apply the technique of [13], we need to make the choices of distances for left and

---

[3]In fact, the reason is simpler than in [7]: every minimal cut (separating s from t) corresponds to a partition of $V^{s,t}$, bounding the number of minimal cuts by 2^k.

right terminals independent. In other words, we need to restrict every left or right terminal r to a fixed set of distances $\Lambda(r)$ such that for every pair $(s,t) \in P$ we have $k + \ell \leq D(s,t)$ for *every* pair of distances $(k, \ell) \in \Lambda(s) \times \Lambda(t)$, while still preserving a large $(1/\mathrm{polylog}(n))$ LP value for these pairs in Constraint (4.2). We show that this can be done via a subtle pruning argument which first applies a standard bucketing technique, and then takes every left terminal s with a current set of possible distances $\lambda_1 < \ldots < \lambda_q$ and applies a non-uniform averaging to show that there is a prefix set of distances $\{\lambda_1, \lambda_2, \ldots, \lambda_{q'}\}$ which are simultaneously good for many terminal pairs involving s. The process is then repeated for right terminals. Details appear in Appendices 5 and D.

5 Junction Tree Algorithm for Pairwise Spanner

This section is a work in progress. For now, we're keeping a special subsection for the Chekuri et al. lemmas, so we can work on those in parallel.

5.1 Lemmas from [13] and [24]

The following lemma is from [13]: ≪**Eden's Comment 5.1:** Needs a proof sketch≫   EC 5.1

Lemma 5.1 (Height reduction) *Let $G = (V, E)$ be an edge-weighted directed graph with edge weights $w : E \to \mathbb{R}_0^+$, 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 $\hat{w} : E(\hat{T}_r) \to \mathbb{R}_0^+$, and a vertex mapping $\Psi : V(\hat{T}_r) \to V(G)$, such that*

- *For any arborescence $J \subseteq G$ 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 $\mathcal{L}(J)$ and $\mathcal{L}(\hat{J})$ be the set of leaves of J and $\hat{J}$, respectively, we have $\mathcal{L}(J) = \Psi(\mathcal{L}(\hat{J}))$. Moreover, $w(J) \leq O(\sigma |\mathcal{L}|^{1/\sigma}) \cdot \hat{w}(\hat{J})$.*

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

The following lemma is from [24]: ≪**Eden's Comment 5.2:** Can we cite a specific lemma/theorem?≫ EC 5.2

Lemma 5.2 (Group Steiner Tree rounding in a tree) *Given an edge-weighted undirected tree T rooted at r, with edge weights $w : E(T) \to \mathbb{R}_0^+$, a collection of vertex sets $\mathcal{S} \subseteq \mathcal{P}(V(T))$, and a solution to the following LP:*

$$\min \quad \sum_{e \in E} w(e) x_e$$
$$\text{s.t.} \quad \text{capacities } \{x_e\} \text{ support one unit of flow from } r \text{ to } S \qquad \forall S \in \mathcal{S}$$
$$x_e \geq 0 \qquad \forall e \in E$$

Then we can efficiently find a subtree $T' \subseteq T$ rooted at r such that for every $S \subseteq \mathcal{S}$ at least one vertex of S participates in the tree T', and $w(T') \leq O(\log n \log |\mathcal{S}| \cdot \sum_{e \in E} w(e) x_e)$.

5.2 Everything else

In this section, we present our approximation algorithm for approximating Minimum Density Junction Tree for the PAIRWISE SPANNER problem, which we then use as part of Algorithm 3 in Section 4 (replacing Algorithm 3 from Section 2). Formally, we prove the following theorem:

Theorem 5.3 *For any constant $\varepsilon > 0$, there is a polynomial time algorithm which, given an unweighted directed n vertex graph $G = (V, E)$, terminal pairs $P \subseteq V \times V$, and distance bounds $D : P \to \mathbb{N}$ (where $D(s,t) \geq d(s,t)$ 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*

$$\min_{r \in V} \frac{|F|}{|\{(s,t) \in P \mid d_{F,r}(s,t) \leq D(s,t)\}|},$$

  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).

Although we solve the same Minimum Density Junction Tree problem as in PAIRWISE DISTANCE PRESERVER and DIRECTED STEINER FOREST, there are key technical difficulties we run into when trying to find junction trees for PAIRWISE SPANNER. In DIRECTED STEINER FOREST, we do not have to concern ourselves with the length of the paths. In PAIRWISE DISTANCE PRESERVER, we are able to restrict the graph to use only shortest paths. Here we have many choices of source-to-root and root-to-sink paths with various distances, and we have to pick a pair of paths with the right distances. Thus, it is not possible to use the algorithm of [13] for the Minimum Density Junction Tree problem as a black box. While our algorithm follows the same overall structure as theirs, key parts of the algorithm need to be applied in conjunction with a reduction, while others need to be replaced.

In fact, we have two technical issues. First, a standard technique that reduces the Minimum Density Junction Tree problem to a tree instance of the density version of the *Group Steiner Forest* problem (GSF) does not work for us, as it does not keep track of distances. Secondly, the approach of [**?**] is to bucket an LP relaxation for Density GSF, giving a relaxation for GSF itself, which can then be rounded directly using the algorithm of [24]. Here, again, we cannot apply their technique directly, because the bucketing needs to prune the possible distances for different terminals, so that they can be picked independently while still respecting the total distance bound between terminal pairs.

As usual, we will try all possibilities for a root $r \in V$, and approximate a minimum density junction tree rooted at r. For every such choice of r, we start by reducing our problem to an instance of the following problem:

Definition 5.4 *In the MINIMUM DENSITY STEINER LABEL COVER problem we are given a directed graph $G = (V, E)$, nonnegative edge costs $w : E \to \mathbb{R}_0^+$, two collections of disjoint vertex sets $\mathcal{S}, \mathcal{T} \subseteq \mathcal{P}(V)$, a collection of set pairs $P \subseteq \mathcal{S} \times \mathcal{T}$, and for each pair $(S,T) \in P$, a relation $R(S,T) \subseteq S \times T$. The goal is to find a set of edges $F \subseteq E$ minimizing the ratio*

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

10

Our reduction allows us to use the first part of [13] by turning our distance problem into a connectivity problem, albeit a considerably more complicated one. The reduction is as follows: 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 edges

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

Set all edge weights for $e \in E_r$ to $w(e) = 1$. Now for every terminal pair $(s, t) \in P$ with distance bound $D(s, t)$, 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 final graph by G_r. Finally, for every terminal pair (s, t), define terminal sets $S_{s,t} = \{(s^t, -i) \mid i \geq 0\} \cap V(G_r)$ and $T_{s,t} = \{(t^s, j) \mid j \geq 0\} \cap V(G_r)$, and relation $R_{s,t} = \{((s^t, -i), (t^s, j)) \in S_{s,t} \times T_{s,t} \mid i + j \leq D(s, t)\}$.

To use our construction, we need one more, much simpler graph: Let G' be a graph comprised of two graphs G_+ and G_- which are copies of G intersecting only in vertex r. Denote by u_+, u_- the copies of u in G_+, G_-, respectively. The following lemma follows directly from our construction:

Lemma 5.5 *For any $f > 0$, and set of terminal pairs $P' \subseteq P$, there exists an edge set $F \subseteq E(G')$ of size $|F| \leq f$ containing a path of length $\leq D(s, t)$ 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 $((s^t, -i), (t^s, j)) \in R_{s,t}$.*

Thus, to prove Theorem 5.3, it suffices to show that we can achieve an $O(n^\varepsilon)$-approximation for the MINIMUM DENSITY STEINER 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 are now ready to apply the first part of the junction tree algorithm of [13]. We apply the algorithm of Lemma 5.1 to our weighted graph (G_r, w) with parameter (constant) $\sigma > 1/\varepsilon$, and obtain a shallow tree $\hat{T}_r$ with weights $\hat{w}$, and 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

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

Therefore, by Lemma 5.5 and Lemma 5.1, it suffices to show the following:

Lemma 5.6 *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*

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

The rest of this section is devoted to proving Lemma 5.6. In the original junction tree algorithm of [13], with no distance constraints, the application of Lemma 5.1 gave a MINIMUM DENSITY GROUP STEINER FOREST instance, for which they formulated and rounded an LP relaxation. However, our reduction yields a connectivity problem of a much more subtle nature, in which it is not enough to choose representatives of various sets of terminals, but to also choose them in a way that satisfies the relations $\hat{R}(s, t)$. Fortunately, the relations $\hat{R}(s, t)$ have a very specific structure, which still allows us to obtain a polylogarithmic approximation (whereas for more general relations

11

the problem would be considerably harder). Nevertheless, these non-cross-product relations require both a new LP relaxation, and a more subtle rounding. We use the following natural LP relaxation for the problem described in Lemma 5.6:

$$
\min \quad \sum_{e \in E(\hat{T}_r)} \hat{w}(e) x_e
$$

$$
\text{s.t.} \quad \sum_{(s,t) \in P} \sum_{(\hat{s},\hat{t}) \in \hat{R}_{s,t}} y_{\hat{s},\hat{t}} = 1
$$

$$
\sum_{\hat{t}:(\hat{s},\hat{t}) \in R_{s,t}} y_{\hat{s},\hat{t}} \leq z_{\hat{s}} \qquad\qquad \forall (s,t) \in P, \hat{s} \in \Psi^{-1}(S_{s,t})
$$

$$
\sum_{\hat{s}:(\hat{s},\hat{t}) \in R_{s,t}} y_{\hat{s},\hat{t}} \leq z_{\hat{t}} \qquad\qquad \forall (s,t) \in P, \hat{t} \in \Psi^{-1}(T_{s,t})
$$

$$
\text{capacities } x_e \text{ support } z_{\hat{s}} \text{ flow from } \hat{s} \text{ to } \hat{r} \qquad\qquad \forall (s,t) \in P, \hat{s} \in \Psi^{-1}(S_{s,t})
$$

$$
\text{capacities } x_e \text{ support } z_{\hat{t}} \text{ flow from } \hat{r} \text{ to } \hat{t} \qquad\qquad \forall (s,t) \in P, \hat{t} \in \Psi^{-1}(T_{s,t})
$$

$$
y_{\hat{s},\hat{t}} \geq 0 \qquad\qquad \forall (s,t) \in P, (\hat{s},\hat{t}) \in \hat{R}_{s,t}
$$

$$
x_e \geq 0 \qquad\qquad \forall e \in E(\hat{T}_r)
$$

This is easily seen to be a relaxation, by considering, for any tree $T \subseteq T_r$, the following solution: Let $P_T = \{(s,t) \in P \mid \exists (\hat{s},\hat{t}) \in \hat{R}_{s,t} : T \text{ contains an } \hat{s} \rightsquigarrow \hat{t} \text{ path}\}$. Let $x_e = 1/|P_T|$ for every $e \in E(T)$, and $x_e = 0$ otherwise. Then clearly the objective function gives $\sum_{e \in E(\hat{T}_r)} \hat{w}(e) x_e = \hat{w}(T)/|P_T|$ as required. Next, for every $(s,t) \in P_T$, let $(\hat{s},\hat{t}) \in \hat{R}_{s,t}$ be a pair of representatives that is connected by T, and set $z_{\hat{s}} = z_{\hat{t}} = y_{\hat{s},\hat{t}} = 1/|P_T|$, and set all other y and z variables to 0. It is easy to check that all constraints are satisfied by this solution.

Given an optimum solution to the above LP relaxation, our goal is to transform such a solution to an LP relaxation for GROUP STEINER TREE (on the tree $\hat{T}_r$), which can then be rounded using known algorithms without losing too much in the LP value. In particular, for pairs $(s,t) \in P$, we need to prune the sets of representatives $\Psi^{-1}(S_{s,t})$ and $\Psi^{-1}(T_{s,t})$, so that we can choose such representatives independently while still respecting the problem structure. Formally, we need to find representative sets $\tilde{S}_{s,t} \subseteq \Psi^{-1}(S_{s,t})$ and $\tilde{T}_{s,t} \subseteq \Psi^{-1}(T_{s,t})$ such that $\tilde{S}_{s,t} \times \tilde{T}_{s,t} \subseteq \hat{R}_{s,t}$ (so that there is no chance that we will pick a pair of representatives corresponding to a path of total length greater than $D(s,t)$), but also make sure that these representative sets still cover a large LP value. This representative set pruning is accomplished by sorting the representatives of each terminal, and taking all representatives of a terminal up to the median representative. The formal pruning procedure is as follows:

- For all terminal pairs $(s,t) \in P$, define $\gamma_{s,t} := \sum_{(\hat{s},\hat{t}) \in \hat{R}_{s,t}} y_{\hat{s},\hat{t}}$.

- For every terminal pair $(s,t) \in P$, sort the representative sets $\Psi^{-1}(S_{s,t})$ and $\Psi^{-1}(T_{s,t})$ by non-decreasing order of distance labels:

  - Sort $\Psi^{-1}(S_{s,t}) = \{\hat{s}_1, \hat{s}_2, \ldots\}$ such that if $i < j$, where $\Psi(\hat{s}_{i'}) = (s^t, -i)$ and $\Psi(\hat{s}_{j'}) = (s^t, -j)$, then $i' < j'$.
  - Sort $\Psi^{-1}(T_{s,t}) = \{\hat{t}_1, \hat{t}_2, \ldots\}$ such that if $i < j$, where $\Psi(\hat{t}_{i'}) = (t^s, i)$ and $\Psi(\hat{t}_{j'}) = (t^s, j)$, then $i' < j'$.

12

- Choose median prefix sets for all terminals: For every terminal pair $(s,t) \in P$, define

$$\mu(s^t) := \min\left\{ k \mid \sum_{i=1}^{k} \sum_{\hat{t}:(\hat{s}_i,\hat{t})\in\hat{R}_{s,t}} y_{\hat{s}_i,\hat{t}} \geq \gamma_{s,t}/2 \right\}, \text{ and}$$

$$\mu(s^t) := \min\left\{ k \mid \sum_{i=1}^{k} \sum_{\hat{s}:(\hat{s},\hat{t}_i)\in\hat{R}_{s,t}} y_{\hat{s},\hat{t}_i} \geq \gamma_{s,t}/2 \right\},$$

and let

$$\tilde{S}_{s,t} := \{\hat{s}_i \mid i \in [\mu(s^t)]\} \quad \text{and} \quad \tilde{T}_{s,t} := \{\hat{t}_i \mid i \in [\mu(t^s)]\}.$$

As we shall see, the choice of median representative prefix sets automatically guarantees that at least half the LP value is preserved. However, we need to make sure that these sets do not contain any pairs that are disallowed by the spanner constraints (formally, by the terminal relations $\hat{R}_{s,t}$):

Lemma 5.7 *In the above algorithm, for every terminal pair $(s,t) \in P$ we have $\tilde{S}_{s,t} \times \tilde{T}_{s,t} \subseteq \tilde{R}_{s,t}$.*

Proof: Fix some terminal pair $(s,t) \in P$. For the purpose of this proof, let us also define the suffix set $U = \{\hat{s}_i \mid i \in \{\mu(s^t),\ldots,|\Psi^{-1}(S_{s,t})|\}\}$. By our choice of $\mu(s^t)$ and definition of $\gamma_{s,t}$, we have that

$$\sum_{\hat{s}\in U} \sum_{\hat{t}:(\hat{s},\hat{t})\in\hat{R}_{s,t}} y_{\hat{s},\hat{t}} = \gamma_{s,t} - \sum_{i=1}^{\mu(s^t)-1} \sum_{\hat{t}:(\hat{s}_i,\hat{t})\in\hat{R}_{s,t}} y_{\hat{s}_i,\hat{t}} > \gamma_{s,t} - \gamma_{s,t}/2 = \gamma_{s,t}/2.$$

Now, let ℓ be such that $\Psi(\hat{s}_{\mu(s^t)}) = (s^t, -\ell)$, and note by our sorted ordering that for every $\hat{s} \in U$ we have $\Psi(\hat{s}) = (s_t, -\ell')$ for some $\ell' \geq \ell$. Therefore, for any representative $\hat{s} \in U$, for any representative $\hat{t} \in T_{s,t}$ such that $(\hat{s},\hat{t}) \in \hat{R}_{s,t}$ we must have $\Psi(\hat{t}) = (t^s, j)$ for some $j \leq D(s,t) - \ell' \leq D(s,t) - \ell$. Thus, letting $k = \max\{k' \mid \Psi(\hat{t}_{k'}) = (t^s, D(s,t) - \ell)\}$, again by our sorted ordering we have

$$\sum_{i=1}^{k} \sum_{\hat{s}:(\hat{s},\hat{t}_i)\in\hat{R}_{s,t}} y_{\hat{s},\hat{t}_i} \geq \sum_{i=1}^{k} \sum_{\substack{\hat{s}\in U \\ (\hat{s},\hat{t}_i)\in\hat{R}_{s,t}}} y_{\hat{s},\hat{t}_i} = \sum_{\hat{s}\in U} \sum_{\hat{t}:(\hat{s},\hat{t})\in\hat{R}_{s,t}} y_{\hat{s},\hat{t}} \geq \gamma/2.$$

This immediately implies that $k \geq \mu(t^s)$. Moreover, for any $\hat{s} \in \tilde{S}_{s,t}$ and $\hat{t} \in \tilde{T}_{s,t}(\subseteq \{\hat{t}_1,\ldots,\hat{t}_k\})$, we have $\Psi(\hat{s}) = (s^t, -\ell')$ and $\Psi(\hat{t}) = (t^s, j')$ such that $\ell' \leq \ell$ and $j' \leq k = D(s,t) - \ell$, and therefore $\ell' + j' \leq D(s,t)$, which implies that $(\hat{s},\hat{t}) \in \hat{R}_{s,t}$, thus proving the lemma. ∎

With this pruning in place, we can run the remaining part of the classical junction tree algorithm. By the definition of $\gamma_{s,t}$ and the LP constraints, we get that $\sum_{(s,t)\in P} \gamma_{s,t} = 1$. We now bucket the pairs P by their γ values. That is, for all $i \in \{0,1,\ldots,\lceil\log|P|\rceil\}$, define $P_i = \{(s,t) \in P \mid \gamma_{s,t} \in (2^{-i-1}, 2^{-i}]\}$. By a standard argument, there exists some i^* such that $\sum_{(s,t)\in P_{i^*}} \gamma_{s,t} \geq \frac{1}{2(\lceil\log|P|\rceil+1)}$, and so $|P_{i^*}| \geq 2^{i^*}/O(\log n)$. Moreover, for every pair $(s,t) \in P_{i^*}$ we have

$$\sum_{\hat{s}\in\tilde{S}_{s,t}} z_{\hat{s}} \geq \sum_{\hat{s}\in\tilde{S}_{s,t}} \sum_{\hat{t}:(\hat{s},\hat{t})\in\hat{R}_{s,t}} y_{\hat{s},\hat{t}} \geq \gamma_{s,t}/2 \geq 2^{-i^*-2},$$

and similarly

$$\sum_{\hat{t}\in\tilde{T}_{s,t}} z_{\hat{t}} \geq 2^{-i^*-2}.$$

13

Now, scaling our LP solution to $x_e^* := \max\{1, 2^{i^*+2} \cdot x_e\}$, we get a (possibly suboptimal) solution to the following LP:

$$
\begin{aligned}
\min \quad & \sum_{e \in E} w(e) x_e^* \\
\text{s.t.} \quad & \text{capacities } \{x_e^*\} \text{ support one unit of flow from } \hat{r} \text{ to } \tilde{S}_{s,t} && \forall (s,t) \in P_{i^*} \\
& \text{capacities } \{x_e^*\} \text{ support one unit of flow from } \hat{r} \text{ to } \tilde{T}_{s,t} && \forall (s,t) \in P_{i^*} \\
& x_e^* \geq 0 && \forall e \in E
\end{aligned}
$$

By Lemma 5.2, we can round this LP solution, and obtain a tree $T' \subseteq \hat{T}_r$ of weight

$$
w(T') = O(\log^2 n \cdot \sum_{e \in E(\hat{T}_r)} w(e) x_e^*) \leq 2^{i^*} \cdot O(\log^2 n \cdot \sum_{e \in E(\hat{T}_r)} w(e) x_e))
$$

such that for every pair $(s,t) \in P_{i^*}$ at least one vertex $\hat{s} \in \tilde{S}_{s,t}$ and at least one vertex $\hat{t} \in \tilde{T}_{s,t}$ are connected through $\hat{r}$ in the tree T'. Recalling by Lemma 5.7 that such $(\hat{s}, \hat{t})$ pairs also belong to $\hat{R}_{s,t}$, we have the following bound on the "density" of the tree T':

$$
\frac{w(T')}{|\{(s,t) \in P \mid \exists (\hat{s}, \hat{t}) \in \hat{R}_{s,t} : T' \text{ contains an } \hat{s} \rightsquigarrow \hat{t} \text{ path}\}|} \leq \frac{w(T')}{|P_{i^*}|} = \frac{1}{O(\log^3 n \cdot \sum_{e \in E(\hat{T}_r)} w(e) x_e)},
$$

thus rounding our original LP, and proving Lemma 5.6 and in turn Theorem 5.3.

6 Hardness Results for Additive Spanners

In this section we give our hardness results for additive spanners. We begin with an informal overview of the result for +1-spanners, and then discuss how to extend the reduction to +k-spanners for larger stretch values. All details can be found in Appendices B and C.

In all of our reductions we will start from the MIN-REP problem, which was first introduced by [27]. In MIN-REP we are given a bipartite graph $G = (A, B, E)$ where A is partitioned into groups $A_1, A_2, \ldots, A_r$ and B is partitioned into groups $B_1, B_2, \ldots, B_r$, with the additional property that every set A_i and every set B_j has the same size (which we will call $|\Sigma|$ due to its connection to the alphabet of a 1-round 2-prover proof system). This graph and partition induces a new bipartite graph G' called the *supergraph* in which there is a vertex a_i for each group A_i and similarly a vertex b_j for each group B_j. There is an edge between a_i and b_j in G' if there is an edge in G between some node in A_i and some node in B_j. A node in G' is called a supernode, and similarly an edge in G' is called a superedge.

A REP-cover is a set $C \subseteq A \cup B$ with the property that for all superedges $\{a_i, b_j\}$ there are nodes $a \in A_i \cap C$ and $b \in B_j \cap C$ where $\{a, b\} \in E$. We say that $\{a, b\}$ *covers* the superedge $\{a_i, b_j\}$. The goal is to construct a REP-cover of minimum size.

We say that an instance of MIN-REP is a YES instance if $OPT = 2r$ (i.e. a single node is chosen from each group) and is a NO instance if $OPT \geq 2^{\log^{1-\epsilon} n} r$. Kortsarz [27] prove that unless NP $\subseteq$ DTIME($2^{\text{polylog}(n)}$), for any constant $\epsilon > 0$ there is no polynomial-time algorithm that can distinguish between YES and NO instances of MIN-REP. So the problem of distinguishing between these instances is the starting point of our reduction.

14

Intuition for +1-Spanners: In the basic reduction framework, due to [27, 21], we start with a MIN-REP instance, and then for every group we add a vertex (corresponding to the supernode) which is connected to vertices in the group by "connection" edges. We then add an edge between any two supernodes that have a superedge in the supergraph. So there is an "outer" graph corresponding to the supergraph, as well as an "inner" graph which is just the Min-Rep graph itself, and they are connected by connection edges. The basic idea is that if we want stretch 3, the only way to span a superedge is to use a path of length 3 that goes through the MIN-REP instance, in which case the MIN-REP edge that is in this path corresponds to nodes in a valid REP-cover. So if we create many copies of the outer nodes (i.e. of the supergraph), then in a YES instance each copy can be covered relatively cheaply by using 3-paths corresponding to the small REP-cover, while in a NO instance every copy *requires* many edges simply in order to 3-span the superedges. This can be generalized to larger stretch values by changing the connection edges into paths of length approximately $k/2$.

Slightly more formally, suppose that we make x copies of the outer nodes. Then in the basic reduction, in a YES instance we can find a 3-spanner that has total size of approximately $x \cdot 2r$, while in a NO instance every 3-spanner has size at least $x \cdot 2^{\log^{1-\epsilon} n} r$.

This reduction strongly depends on having canonical paths of length at least 3: since there is an $O(\log n)$-approximation for multiplicative stretch 2 spanners [26], no similar reduction can exist. So if we want to prove hardness for +1-spanners, we need to make the true distance between supernodes at least 2, rather than 1. The obvious way to do this would be to subdivide each superedge into a path of length 2. But now consider these new vertices: clearly in the optimal solution they must have degree at least one, and thus even in a YES instance the sparsest spanner must have size at least $x \cdot |E(G')| \approx x \cdot 2^{\log^{1-\epsilon} n} r$. So we have entirely lost the hardness!

The intuitive solution is to subdivide superedges as before, except we use the *same* middle vertex for all of the x copies. Thus we have to add only $|E(G')|$ extra nodes rather than $x \cdot |E(G')|$, and so the fact that these nodes must have degree at least 1 in the optimal solution is no longer a problem. Of course, now we have other problems: how do we span all of the new edges we added? To do this we have to add yet another dummy node and paths of length 2 from each outer node each of the middle superedge nodes, so it does not seem like we have much progress. This seems like a catch-22: every time we add new vertices or edges to span other pairs, it becomes too expensive to span what we've added. But now by carefully hooking these nodes up together, it turns out that we can use the same extra dummy node for each of the x copies, so the total extra cost ends up being only $O(rx)$, which is still small enough that we maintain the hardness gap.

Hardness for +k-Spanners: To extend our hardness reduction to larger stretch values we will want to use the same basic idea: instead of supernodes being at distance 1 from each other if they have a superedge (as in the classical reductions [27, 21, 17]), we will make them further away initially and make sure that canonical paths have length exactly k more than the original lengths. But here the reason we need large initial distance is very different from the reason that we needed it for +1-spanners: when the additive stretch is larger than 1, then including superedges directly would lead to the possibility of spanning a superedge using non-canonical paths made up entirely of superedges. This was the main difficulty in proving hardness of approximation for multiplicative spanners, and was overcome by [17] by sparsifying the superedges so there are no short non-canonical paths (note that allowing directed edges solves this problem, for both multiplicative and additive spanners, but we want hardness for even the undirected setting). But we pay a price in the hardness for doing this sparsification: the hardness drops to $2^{\frac{\log^{1-\epsilon} n}{k}}$, which becomes negligible when $k = \Omega(\log n)$.

Pushing past this boundary requires giving up on sparsifying. Thus the supergraph might have girth 4, so if the original distance between two supernodes connected by a superedge is d, then the spanner instance we construct might have a path of length $3d$ using only these superedge paths. To prevent such paths from being a problem, we will have to make them be too long, i.e. we will need $3d - d = 2d$ to be greater than the additive stretch k. But again we have the same Catch-22 as in $+1$-spanners: if we replace each superedge by a long path, then simply spanning all of the edges in those paths is too expensive. Moreover, it now becomes difficult to span pairs that were innocuous before, e.g. two copies of the same supernode. Overcoming these issues requires adding even more extra paths and vertices, which create their own complications when trying to span them. But these difficulties can be overcome with enough technical work; see Appendix C for details.

References

[1] Amir Abboud and Greg Bodwin. The 4/3 additive spanner exponent is tight. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 351–361. ACM, 2016.

[2] Amir Abboud and Greg Bodwin. Error amplification for pairwise spanner lower bounds. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 841–854. SIAM, 2016.

[3] Donald Aingworth, Chandra Chekuri, Piotr Indyk, and Rajeev Motwani. Fast estimation of diameter and shortest paths (without matrix multiplication). *SIAM J. Comput.*, 28(4):1167–1181, 1999.

[4] Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9(1):81–100, 1993.

[5] Sanjeev Arora and Carsten Lund. Approximation algorithms for np-hard problems. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, chapter Hardness of Approximations, pages 399–446. PWS Publishing Co., Boston, MA, USA, 1997.

[6] Surender Baswana, Telikepalli Kavitha, Kurt Mehlhorn, and Seth Pettie. Additive spanners and (α, β)-spanners. *ACM Trans. Algorithms*, 7(1):5:1–5:26, December 2010.

[7] Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and directed steiner forest. *Inf. Comput.*, 222:93–107, 2013.

[8] Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 932–941, 2009.

[9] Greg Bodwin and Virginia Vassilevska Williams. Better distance preservers and additive spanners. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 855–872. SIAM, 2016.

[10] Shiri Chechik. New additive spanners. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 498–512. SIAM, 2013.

[11] Shiri Chechik. Additive spanners. In *Encyclopedia of Algorithms*, pages 22–24. 2016.

[12] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM Journal on Computing*, 39(5):1772–1798, 2010.

[13] Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Trans. Algorithms*, 7(2):18, 2011.

[14] E. Chlamtác, M. Dinitz, and R. Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 758–767, 2012.

[15] Eden Chlamtác and Michael Dinitz. Lowest Degree k-Spanner: Approximation and Hardness. In *Proceedings of APPROX/RANDOM*, volume 28 of *LIPIcs*, pages 80–95, 2014.

[16] Don Coppersmith and Michael Elkin. Sparse sourcewise and pairwise distance preservers. *SIAM J. Discrete Math.*, 20(2):463–501, 2006.

[17] Michael Dinitz, Guy Kortsarz, and Ran Raz. Label cover instances with large girth and the hardness of approximating basic k-spanner. *ACM Trans. Algorithms*, 12(2):25, 2016.

[18] Michael Dinitz and Robert Krauthgamer. Directed spanners via flow-based linear programs. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 323–332, 2011.

[19] Michael Dinitz and Robert Krauthgamer. Fault-tolerant spanners: better and simpler. In *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, 2011.

[20] Michael Dinitz and Zeyu Zhang. Approximating low-stretch spanners. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 821–840, Philadelphia, PA, USA, 2016. Society for Industrial and Applied Mathematics.

[21] Michael Elkin and David Peleg. The hardness of approximating spanner problems. In *STACS*, pages 370–381, 2000.

[22] Paul Erdős. Extremal problems in graph theory. *Theory Graphs Appl., Proc. Symp. Smolenice 1963*, pages 29–36, 1964.

[23] Moran Feldman, Guy Kortsarz, and Zeev Nutov. Improved approximation algorithms for directed steiner forest. *J. Comput. Syst. Sci.*, 78(1):279–292, 2012.

[24] Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000.

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

[26] G. Kortsarz and D. Peleg. Generating sparse 2-spanners. *J. Algorithms*, 17(2):222–236, 1994.

[27] Guy Kortsarz. On the hardness of approximating spanners. *Algorithmica*, 30(3):432–450, 2001.

[28] David P. Woodruff. Lower bounds for additive spanners, emulators, and more. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 389–398. IEEE Computer Society, 2006.

[29] David P. Woodruff. Additive spanners in nearly quadratic time. In *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming*, ICALP'10, pages 463–474, Berlin, Heidelberg, 2010. Springer-Verlag.

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

A Directed Steiner Forest with uniform costs

We show here, in more detail, how our algorithm for preservers may be adapted for unweighted Directed Steiner Forest, to give an $n^{3/5+\varepsilon}$-approximation. We start with our adaptation of Algorithm 1, for the corresponding case when $\mathsf{OPT} \geq n^{4/5}$. As mentioned in Section 3, we use the following standard LP relaxation:

$$
\begin{aligned}
\min \quad & \textstyle\sum_{e \in E} x_e \\
\text{s.t.} \quad & \text{capacities } x_e \text{ support a one unit } s - t \text{ flow } f_{s,t} && \forall (s,t) \in P \\
& x_e \geq 0 && \forall e \in E
\end{aligned}
$$

For every pair $(s,t) \in P$, we define its local graph w.r.t. this LP: let $V^{s,t}$ be the set of vertices involved in the flow $f_{s,t}$. We can define k-thick and k-thin terminal pairs accordingly – (s,t) is a k-thick pair if $|V^{s,t}| \geq k$, and otherwise it is k-thin. The guarantee of Lemma 2.3 follows here with the same analysis. The only difference is the justification of Claim 2.1 from [7]. In their paper, the proof of this claim relies entirely on showing that the number of minimal "antispanners" (edge sets whose removal from the flow $f_{s,t}$ increases the distance from s to t) is bounded by $|E| \cdot |V^{s,t}|^{|V^{s,t}|}$. In our case, the proof is even simpler, since rather bounding the number of antispanners, we need to bound the number of minimal cuts, which is clearly at most $2^{V^{s,t}}$ (since every minimal cut is simply a partition).

Now suppose $\mathsf{OPT} < n^{4/5}$. As discussed in the proof of Lemma 2.6, as long as we have an algorithm to find a junction tree (or really any edge set) with density at most $\alpha \cdot \mathsf{OPT}/|P|$, we can repeatedly apply such an algorithm to the set of remaining unconnected pairs, adding at most $O(\alpha \cdot \mathsf{OPT})$ edges overall. One way to find an edge set with good density is indeed via junction trees, or by adapting one of our other algorithms to connect a constant fraction of pairs. Thus, we need an algorithm which finds an edge set with density at most $n^{3/5+\varepsilon} \cdot \mathsf{OPT}/|P|$. Rather than using buckets as in Section 2, we use thresholding. While our approach at this point is very similar to that of [7], the actual parameters and combination of algorithmic components is slightly different, so we give the algorithm and analysis here for completeness. Set $D = n^{1/5}$, and define a layered graph $G_D = (V_D, E_D)$ where $V_D = V \times \{0, 1, \dots, D\}$, and

$$
E_D = \{((u, j-1), (v, j)) \mid j \in [D] \wedge (((u,v) \in E) \vee (u = v \in V))\}
$$

We start with the following LP relaxation:

$$
\begin{aligned}
\min \quad & \sum_{e \in E} x_e \\
\text{s.t.} \quad & f_{s,t} \text{ is a } (s,0) - (t,D) \text{ flow} & & \forall (s,t) \in P & & \text{(A.4)} \\
& \sum_{j=1}^{D} f_{s,t}(e) \leq x_e & & \forall e \in E (s,t) \in P & & \text{(A.5)} \\
& |f_{s,t}| \leq 1 & & \forall (s,t) \in P & & \text{(A.6)} \\
& \sum_{(s,t) \in P} |f_{s,t}| \geq |P|/2 & & & & \text{(A.7)} \\
& x_e \geq 0 & & \forall e \in E & & \text{(A.8)}
\end{aligned}
$$

While this LP seems to increase the number of vertices by a factor of D, it is simply a compact way of formulating a flow LP on G where the flows are restricted to paths of length at most D, and we may think of the flows $f_{s,t}$ as such.

The algorithm now trades off a slight variant of Algorithm 2 and Algorithm 3, and chooses the sparser of the two edge sets:

Algorithm 4

- Solve the above LP relaxation (if feasible), and run Algorithm 2 on pairs $P_{1/4} := \{(s,t) \in P \mid |f_{s,t}| \geq 1/4\}$, where $V^{s,t}$ is the support (in V) of flow $f_{s,t}$, and distance bounds D (w.r.t. the hitting set), and let F_0 be the edge set found by this algorithm.

- For every $u \in V$, use the algorithm of [13] to find an approximately optimal junction tree in graph G and all pairs P, and let F^u and P^u be the set of edges used and terminals connected by this algorithm, respectively.

- If the LP is infeasible, or $|F_0|/|P| > \min_u |F_u|/|P_u|$, then add output F_u that minimizes this ratio. Otherwise, output F_0.

The following lemma gives the required guarantee:

Lemma A.1 *Assuming* $\mathsf{OPT} < n^{4/5}$, *Algorithm 4 outputs an edge set F' which connects terminal pairs $P' \subseteq P$ such that $|F'|/|P'| \leq n^{3/5+\varepsilon} \cdot \mathsf{OPT}/|P|$.*

Proof: For some optimum solution, let P_D be the set of terminal pairs in P which the solution connects using paths of length at most D. Consider two cases.

Case 1: $|P_D| \geq |P|/2$.
In this case, the above LP is feasible, and has value at most OPT. Also note that constraints (A.6) and (A.7) imply that $|P_{1/4}| \geq |P|/3$, and so by the same analysis as Lemma 2.5, the algorithm connects these pairs using at most $\tilde{O}(n^{3/5}\mathsf{OPT})$ edges, giving a set of density $\tilde{O}(n^{3/5} \cdot \mathsf{OPT}/|P|)$.

Case 2: $|P_D| < |P|/2$.
In this case, at least $P/2$ pairs are connected by paths of length at least D, and so we find a junction tree with density at most $n^{\varepsilon} \cdot \mathsf{OPT}^2/(|D||P|) < n^{3/5+\varepsilon} \cdot \mathsf{OPT}/|P|$ by the exact same analysis as in the proof of Lemma 2.6. ∎

B   Hardness Results for +1-Spanners

In all of our reductions we will start from the MIN-REP problem, which was first introduced by [27]. In MIN-REP we are given a bipartite graph $G = (A, B, E)$ where A is partitioned into groups $A_1, A_2, \ldots, A_r$ and B is partitioned into groups $B_1, B_2, \ldots, B_r$, with the additional property that every set A_i and every set B_j has the same size (which we will call $|\Sigma|$ due to its connection to the alphabet of a 1-round 2-prover proof system). This graph and partition induces a new bipartite graph G' called the *supergraph* in which there is a vertex a_i for each group A_i and similarly a vertex b_j for each group B_j. There is an edge between a_i and b_j in G' if there is an edge in G between some node in A_i and some node in B_j. A node in G' is called a supernode, and similarly an edge in G' is called a superedge.

A REP-cover is a set $C \subseteq A \cup B$ with the property that for all superedges $\{a_i, b_j\}$ there are nodes $a \in A_i \cap C$ and $b \in B_j \cap C$ where $\{a, b\} \in E$. We say that $\{a, b\}$ *covers* the superedge $\{a_i, b_j\}$. The goal is to construct a REP-cover of minimum size.

We say that an instance of Min-Rep is a YES instance if $OPT = 2r$ (i.e. a single node is chosen from each group) and is a NO instance if $OPT \geq 2^{\log^{1-\epsilon} n} r$.

Theorem B.1 ([27]) *Unless* $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$, *for any constant $\epsilon > 0$ there is no polynomial-time algorithm that can distinguish between YES and NO instances of* MIN-REP. *This is true even when the graph and the supergraph are regular, and both the supergraph degree and $|\Sigma|$ are polynomial in $2^{\log^{1-\epsilon} n}$.*

B.1   The Reduction

Suppose we are given a MIN-REP instance $\widetilde{G} = (A, B, \widetilde{E})$ with associated supergraph $G' = (U, V, E')$. For any vertex $w \in U \cup V$ we let $\Gamma(w)$ denote its group. So $\Gamma(u) \subseteq A$ for $u \in U$, and $\Gamma(v) \subseteq B$ for $v \in V$. We will assume without loss of generality that G' is regular with degree $d_{G'}$ and $\widetilde{G}$ is regular with degree $d_{\widetilde{G}}$. Let $x \in \mathbb{N}$ be a parameter which we will set later.

Our +1-spanner instance will have several kinds of vertices. We first define the following vertex sets:

$$V_{out}^L = U \times [x] \qquad\qquad V_{out}^R = V \times [x]$$
$$S = \{s_y : y \in U \cup V\} \qquad\qquad M = \{m_{uv} : \{u, v\} \in E'\}.$$

In other words, the V_{out} vertices are just x copies of the nodes in the supergraph, S consists of one additional "special" node for each node in the supergraph, and M has a vertex for each superedge (these are the "middle" nodes). Let $V_R = V_{out}^L \cup V_{out}^R \cup A \cup B \cup S \cup M$ be the "main" vertex set. For technical reasons these vertices will not quite be enough, so we will also define a single special node t and a set of nodes $T = \{t_y : y \in V_R\}$. The final vertex set of our instance will be

$$V_G = V_R \cup \{t\} \cup T = V_{out}^L \cup V_{out}^R \cup A \cup B \cup S \cup M \cup \{t\} \cup T.$$

Now that our vertices are defined, we need to define edges. We first add inner edges, which will just be a copy of the Min-Rep instance $\widetilde{G}$. Formally, since $A, B \subset V_G$, we just let $E_{in} = \widetilde{E}$. We will next connect the outer nodes to the inner nodes using connection edges:

$$E_{con} = \{\{(u, i), a\} : u \in U \cup V \wedge a \in \Gamma(u) \wedge i \in [x]\},$$

The next set of edges (the outer edges) form length-2 paths for each superedge:

$$E_{out} = \{\{(u, i), m_{uv}\} : u \in U \wedge i \in [x] \wedge \{u, v\} \in E'\},$$
$$\bigcup \{\{(v, j), m_{uv}\} : v \in V \wedge j \in [x] \wedge \{u, v\} \in E'\}.$$

We next connect nodes in S using three edge sets: one which connects S to the outer nodes, one which connects S to the inner nodes, and one which connect S to the middle nodes M:

$$E_{so} = \{\{(u, i), s_u\} : u \in U \cup V \wedge i \in [x]\},$$
$$E_{si} = \{\{a, s_u\} : u \in U \cup V \wedge a \in \Gamma(u)\}$$
$$E_{sm} = \{\{s_u, m_{uv}\} : u \in U \wedge \{u, v\} \in E'\} \bigcup \{\{s_v, m_{uv}\} : v \in V \wedge \{u, v\} \in E'\}.$$

We next add group edges to form a clique inside each group:

$$E_{group} = \{\{a, b\} : \Gamma^{-1}(a) = \Gamma^{-1}(b)\}.$$

Finally, we will add "star" edges to make the entire graph have diameter 4 by going through the special node t:

$$E_{star} = \{\{t, t_y\} : y \in V_R\} \bigcup \{\{t_y, y\} : y \in V_R\}.$$

Our final edge set E_G is the union of all of these sets: $E_G = E_{in} \cup E_{con} \cup E_{out} \cup E_{so} \cup E_{si} \cup E_{sm} \cup E_{group} \cup E_{star}$.

B.2 Analysis

B.2.1 Soundness

We first analyze the soundness of this reduction: we want to show that the size any +1-spanner is lower bounded by OPT_{MR} (the optimal solution of the Min-Rep instance $\tilde{G}$). Let H be an arbitrary +1-spanner of $G = (V_G, E_G)$.

Definition B.2 *A path between outer nodes (u, i) and (v, j) is called* canonical *if it has the form $(u, i) \rightarrow a \rightarrow b \rightarrow (v, j)$, where $a \in \gamma(u)$ and $b \in \Gamma(v)$. In other words, it is a length 3 path whose first and last edges are from E_{con} and whose middle edge is from E_{in}.*

Note that if $\{u, v\}$ is a superedge then (u, i) and (v, j) are at distance 2 in G because of the outer edges. Thus in H they must be at distance 2 or 3. The intuition is that if $\{u, v\}$ is a superedge then H should contain a canonical path from (u, i) to (v, j) for all $i, j \in [x]$. Such a path exists, since it corresponds to choosing nodes in a REP-cover to satisfy the $\{u, v\}$ superedge. Unfortunately this isn't quite true, but it's "true enough" – with only a small loss, we can guarantee that *enough* paths are canonical.

For the purposes of lower bounding $|E(H)|$, it will turn out to be easier to only consider the case of $j = i$. Note that (u, i) and (v, j) are also at distance 2 when $j \neq i$, but for the purpose of the analysis we will only care about $j = i$. Thus we will not try to make all possible paths canonical, but rather only when $j = i$

Definition B.3 *A +1-spanner H' of G is* canonical *if for all outer nodes (u, i) and (v, i) where $\{u, v\} \in E'$ and $i \in [x]$, there is a canonical path between (u, i) and (v, i).*

Lemma B.4 *There is a canonical +1-spanner H' of G with $|E(H')| \leq 4|E(H)|$.*

Proof: For each superedge $\{u, v\} \in E'$, let $a_{uv} \in \Gamma(u)$ and $b_{uv} \in \Gamma(v)$ be arbitrary inner nodes so that $\{a_{uv}, b_{uv}\} \in \widetilde{E}$, i.e. so that there is a canonical path $(u, i) \to a_{uv} \to b_{uv} \to (v, j)$ for all $i, j \in [x]$. We create a new graph H' by starting with H, and then for each $((u, i), (v, i))$ pair of outer nodes where $\{u, v\} \in E'$ and $i \in [x]$ but where there is no canonical path from (u, i) to (v, i), we add the connection edges $\{(u, i), a_{uv}\}$ and $\{(v, i), b_{uv}\}$ and the inner edge $\{a_{uv}, b_{uv}\}$ (if these edges do not already exist). Since H' contains H, we know that H' is also a +1-spanner, and it is canonical by construction.

So it remains to prove the size bound. How many edges did we add to H to get H'? Suppose we added (at most 3) edges to canonically span the pair $((u, i), (v, i))$. Then in H the path between (u, i) and (v, i) already had length at most 3, but was not a canonical path. Since it was not a canonical path but had length 3, it must have one of the following forms:

1. $(u, i) \to m_{uv} \to (v, i)$, or

2. $(u, i) \to m_{uv} \to s_v \to (v, i)$, or

3. $(u, i) \to s_u \to m_{uv} \to (v, i)$.

This classification is easy to see by inspection. The unique path of type 1 is clearly the only path of length 2. And the only nodes reachable in 2 hops from (u, i) are inner nodes, outer nodes of the form (u, j), s_u, $m_{uv'}$ (where v' is not necessarily equal to v), $s_{v'}$, $t_{(u,i)}$, and t. If a 2-hop path to an inner node can be extended in one hop to (v, i) then we have a canonical path, so by assumption no such path exists. Of the other nodes reachable in two hops, the only ones adjacent to (v, i) are m_{uv} and s_v. Hence no paths other than paths 2 and 3 exist of length 3.

This classification implies that either $\{(u, i), m_{uv}\}$ or $\{(v, i), m_{ab}\}$ (or both) is an edge in $E(H)$. Let e_{uvi} be one of these edges (if both exist, choose one arbitrarily). When creating H', we added up to three edges in order to add a canonical path between (u, i) and (v, i). We will charge those edges to e_{uvi}.

When we do this for all $\{u, v\} \in E'$ and $i \in [x]$, it is clear that we have charged a different edge for each such (superedge, value) pair. Since each charge involves at most three new edges, and every existing edge is charged at most once, we immediately get the lemma. ∎

Theorem B.5 *Any +1-spanner H of G has $|E(H)| \geq x \cdot OPT_{MR}/4$.*

Proof: By Lemma B.4, we just need to show that $|E(H')| \geq x \cdot OPT_{MR}$.

For $i \in [x]$, let $C_i \subseteq A \cup B$ be the set of inner nodes which in H' are adjacent to the associated outer node, i.e. $C_i = \{a \in A \cup B : \{(\Gamma^{-1}(a), i), a\} \in E(H')\}$. We first claim that C_i is a REP-cover of $\widetilde{G}$ for all $i \in [x]$. To see this, fix $i \in [x]$ and a superedge $\{u, v\} \in E'$. Since H' is canonical there is a canonical path $(u, i) \to a \to b \to (v, i)$ in H'. Thus a and b are both in C_i, and since the middle edge of the canonical path is an edge in $\widetilde{E}$ we know that C_i covers the superedge $\{u, v\}$. Hence C_i is a valid REP-cover for all $i \in [x]$, and thus $|C_i| \geq OPT_{MR}$

Now note that by definition, every node $a \in C_i$ is incident on an edge $\{a, (\Gamma^{-1}(a), i)\}$. Since this edge is different for all $i \in [x]$ and $a \in C_i$, we immediately get that $|E(H')| \geq \sum_{i=1}^{x} |C_i| \geq x \cdot OPT_{MR}$. ∎

B.2.2 Completeness

We now show completeness: that there is a $+1$-spanner of G which does not cost too much *more* than OPT_{MR} in the YES case. Recall that in a YES instance there is a valid REP-cover of size $|U| + |V|$ where for each supernode we choose exactly one representative. Let C be such a REP-cover, for each $u \in U$ let a_u be the unique element of $C \cap \Gamma(u)$, and for each $v \in V$ let b_v be the unique element of $C \cap \Gamma(v)$.

The subgraph H of G we will analyze will include $E_{in} \cup E_{so} \cup E_{si} \cup E_{sm} \cup E_{star}$. It will also include connection edges $\{(u, i), a_u\}$ for all $i \in [x]$ and $u \in U$ and connection edges $\{(v, i), b_v\}$ for all $i \in [x]$ and $v \in V$. Finally, it will include a star of group edges in each group, with the representative from C as the center. More formally, for each $u \in U$ we will include the group edges $\{\{a_u, a\} : a \in \Gamma(u)\}$ and for each $v \in V$ we will include the group edges $\{\{b_v, b\} : b \in \Gamma(v)\}$.

We first analyze the size of H, and then later will prove that it is a $+1$-spanner of G. Recall that $d_{G'}$ is the degree of the supergraph G' (which is regular). Let n' be the number of nodes in the supergraph, and let $|\Sigma|$ denote the size of each group in $\widetilde{G}$.

Theorem B.6 *H has at most $8n'd_{G'}|\Sigma|^2 + 4n'x$ edges when $\widetilde{G}$ is a YES instance.*

Proof: Clearly $E_{in} = \widetilde{E}$ has at most $n'd_{G'}|\Sigma|^2$ edges, E_{so} has at most $n'x$ edges, E_{si} has at most $n'|\Sigma|$ edges, E_{sm} has at most $n'd_{G'}$ edges, and E_{star} has at most $2|V_R| = 2(n'x + n' + n'd_{G'})$ edges The number of group edges added is clearly at most $n'|\Sigma|$, and the number of connection edges is at most $n'x$. Putting this together, we get that

$$
\begin{aligned}
|E(H)| &\leq n'd_{G'}|\Sigma|^2 + n'x + n'|\Sigma| + n'd_{G'} + 2(n'x + n' + n'd_{G'}) + n'|\Sigma| + n'x \\
&\leq 8n'd_{G'}|\Sigma|^2 + 4n'x
\end{aligned}
$$

as claimed. ∎

Theorem B.7 *H is a $+1$-spanner of G.*

Proof: Since H includes all star edges it has diameter at most 4, and hence any two vertices which were originally at distance 3 or 4 are spanned by H. Thus we only need to analyze pairs which were at distance 1 or 2 in G.

We begin by analyzing pairs which were at distance 1 in G, i.e. edges of G. If some edge is also in H then it is trivially spanned. The only edges of G not in H are the outer edges, some group edges, and some connection edges. Each outer edge has a length 2 path spanning it by using one edge of E_{so} and one edge of E_{sm}. Each group edge has a length 2 path spanning it using the star inside every group that is included in H. Similarly, each connection edge has a 2-path spanning it using the included connection edge and an included star edge. Slightly more formally, consider a connection edge $\{(u, i), a\}$. Then H includes the connection edge $\{(u, i), a_u\}$ and the group edge $\{a_u, a\}$. The same analysis holds for connection edges on the other side.

We now consider pairs which were at distance 2 in G, and prove that they are at distance at most 3 in H. To do this, first note that if the 2-path connecting two nodes in G has both edges in H then it is trivially still spanned in H, and if one of the two edges are in H then it still has distance at most 3 in H (since by the previous analysis all missing edges are replaced by a path of length 2). So we only need to worry about pairs of vertices at distance 2 in G where both edges on the 2-path in G are not in H.

23

This leaves only a few cases to analyze, based on the types of the two edges in the path. We consider them each.

1. Two outer edges. From an outer node $(u, i) \in V_{out}^L$, any path involving two outer edges must end at either an outer node (v, j) where $\{u, v\} \in E'$ or at a node (u, j) with $j \neq i$. For the first of these, we know by construction that there is a length-3 canonical path from (u, i) to (v, j). For the second, there is still a 2-path by going through s_u and using two edges from E_{so}. The same analysis holds for paths from outer nodes $(v, j) \in V_{out}^R$.

   The only other type of 2-path involving only outer edges are path between two middle nodes, i.e. 2-paths from m_{uv} to $m_{u'v'}$. For such a path to exist, either $u' = u$ or $v' = v$, and hence H still has a 2-path of the form $m_{uv} \to s_u \to m_{uv'}$ or of the form $m_{uv} \to s_v \to m_{u'v}$.

2. Two connection edges. Any 2-path involving two missing connection edges must be either between outer nodes (u, i) and (u, j) with $i \neq j$, or between two inner nodes a, b with $\Gamma^{-1}(a) = \Gamma^{-1}(b)$. In the first case there is still a 2-path between the nodes by using edges in E_{so}, and in the second case there is still a 2-path by using the remaining edges of E_{group} in $\Gamma(\Gamma^{-1}(a))$.

3. Two group edges. This is trivial since any pair of nodes connected by such a 2-path are also connected by a single group edge, so the analysis for a single edge holds.

4. Outer edge and connection edge. Any such path is from a middle node m_{uv} to an inner node $a \in \Gamma(u) \cup \Gamma(b)$. A 2-path between these nodes exists in H by using an edge in E_{sm} and an edge in E_{si}.

5. Outer edge and group edge. No such path exists.

6. Connection edge and group edge. The two endpoints of such a path must be an outer node (u, i) and an inner node $a \in \Gamma(u)$. But G already has a connection edge between them, so they are at distance 1 in G and the previous analysis applies.

   Thus H is a +1-spanner of G. ∎

B.2.3 Putting it Together

We can now finish the proof of hardness.

Theorem B.8 *For any constant $\epsilon > 0$, there is no polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm for the +1-spanner problem unless $NP \subseteq DTIME(2^{\mathrm{polylog}(n)})$.*

Proof: We first instantiate x to be $d_{G'}|\Sigma|^2$. With this setting of x, it is obvious that the graph G created by our reduction has size n which is polynomial in the size $n'|\Sigma|$ of the Min-Rep instance $\widetilde{G}$ (and that we can create G in polynomial time).

The standard Min-Rep analysis implies that under the given complexity assumption, there is no polynomial-time algorithm which can distinguish between the YES case where $OPT_{MR} = n'$ and the NO case where $OPT_{MR} \geq n' 2^{\log^{1-\epsilon}(n'|\Sigma|)}$. Let OPT denote the size of the optimal +1-spanner for G. Then Theorem B.5 implies that if $\widetilde{G}$ is a NO instance then $OPT \geq x \cdot OPT_{MR}/4 \geq xn' \cdot 2^{\log^{1-\epsilon}(n'|\Sigma|)} / 4$. On the other hand, if $\widetilde{G}$ is a YES instance, then Theorems B.6 and B.7 imply that $OPT \leq 12n'x$. Thus we cannot approximate the +1-spanner problem better than $48 \cdot 2^{\log^{1-\epsilon}(n|\Sigma|)}$. By using a sufficiently smaller value of ϵ (and restricting our attention to large enough input instances), this implies hardness of $2^{\log^{1-\epsilon} n}$ as claimed. ∎

C   Hardness for $+k$-Spanners

We now prove hardness for larger additive stretch, with the goal of proving Theorem 1.11.

C.1   The Reduction

Suppose we are given a bipartite Min-Rep instance $\widetilde{G} = (A, B, \widetilde{E})$ with associated supergraph $G' = (U, V, E')$. For any vertex $w \in U \cup V$ we let $\Gamma(w)$ denote its group. So $\Gamma(u) \subseteq A$ for $u \in U$, and $\Gamma(v) \subseteq B$ for $v \in V$. We will assume without loss of generality that G' is regular with degree d, and let $n' = |U \cup V|$. Let $x \in \mathbb{N}$ be a parameter which we will set later.

Our $+k$-spanner instance will have several kinds of vertices. We first define the following vertex sets:

$$V_{out}^L = U \times [x] \times [k-1] \qquad\qquad V_{out}^R = V \times [x] \times [k-1]$$
$$S = \{s_y : y \in U \cup V\} \qquad\qquad L = \{\ell_{u,i,j} : u \in U \cup V, i \in [x], j \in [k-1]\}$$
$$M = \{m_{uv}^i : \{u, v\} \in E' \wedge i \in [k-2]\} \qquad V_{in} = A \cup B$$
$$Q = \{q_y : y \in U \cup V\} \qquad\qquad P = \left\{p_{u,i,j} : u \in U \cup V, i \in [x], j \in \left[\left\lceil \frac{k-1}{2} \right\rceil\right]\right\}$$

Let the union of the above vertices be $V_R = V_{out}^L \cup V_{out}^R \cup A \cup B \cup S \cup M \cup L \cup P \cup Q$, which will be the "main" vertex set. Note that $|V_R| \le kn'x + n'|\Sigma| + n' + n'kx + n'dk + n' + n'kx \le 3n'kx + n'(|\Sigma| + 2 + dk) \le 3n'kx + 2n'(|\Sigma| + dk)$

In order to decrease the diameter, we will also have an extra node t and extra nodes

$$T = \{t_{y,i} : y \in V_R \wedge i \in [k-1]\}.$$

The final vertex set of our instance will be

$$V_G = V_R \cup T = V_{out}^L \cup V_{out}^R \cup A \cup B \cup S \cup M \cup L \cup Q \cup P \cup T \cup \{t\}.$$

Now that our vertices are defined, we need to define edges. We first add inner edges, which will just be a copy of the Min-Rep instance $\widetilde{G}$. Formally, since $A, B \subset V_G$, we just let $E_{in} = \widetilde{E}$.

We next define the crucial connection edges, we connect the inner nodes to the outer nodes:

$$E_{con} = \{\{(u, i, 1), a\} : u \in U \cup V \wedge a \in \Gamma(u) \wedge i \in [x]\}.$$

We now need to define paths among the outer nodes:

$$E_{path} = \{\{(u, i, j), (u, i, j+1)\} : u \in U \cup V \wedge i \in [x] \wedge j \in [k-2]\}$$

The next edges also form paths: E_P is set up to provide alternate bounded-length paths between the outermost outer nodes, and E_L is set up to provide bounded-length paths from the outermost outer nodes to S.

$$E_L = \{\{(u, i, k-1), \ell_{u,i,1}\} : u \in U \cup V \wedge i \in [x]\}$$
$$\cup \{\ell_{u,i,j}, \ell_{u,i,j+1}\} : u \in U \cup V \wedge i \in [x] \wedge j \in [k-2]\}$$

$$\cup \{\{\ell_{u,i,k-1}, s_u\} : u \in U \cup V \wedge i \in [x]$$
$$E_P = \{\{(u,i,k-1), p_{u,i,1}\} : u \in U \cup V \wedge i \in [x]\}$$
$$\cup \{\{p_{u,i,j}, p_{u,i,j+1}\} : u \in U \cup V \wedge i \in [x] \wedge j \in \left\lceil \frac{k-1}{2} \right\rceil - 1\}$$
$$\cup \{\{p_{u,i,\lceil \frac{k-1}{2} \rceil}, q_u\} : u \in U \cup V \wedge i \in [x]\}$$

We now connect the nodes in S using two edge sets:

$$E_{so} = \{\{s_u, \ell_{u,i,k-1}\} : u \in U \cup V \wedge i \in [x]\}$$
$$E_{sm} = \{\{s_u, m_{uv}^1\} : u \in U \wedge \{u,v\} \in E'\} \cup \{\{s_v, m_{uv}^{k-2}\} : v \in V \wedge \{u,v\} \in E'\}.$$

And now we connect the nodes in M via more paths. Note that the following edge set is empty if $k = 3$.

$$E_M = \{\{m_{uv}^i, m_{uv}^{i+1}\} : \{u,v\} \in E' \wedge i \in [k-3]\}.$$

We connect M to V_{out} using outer edges:

$$E_{out} = \{\{(u,i,k-1), m_{uv}^1\} : u \in U \wedge i \in [x] \wedge \{u,v\} \in E'\}$$
$$\cup \{\{(v,i,k-1), m_{uv}^{k-2}\} : v \in V \wedge i \in [x] \wedge \{u,v\} \in E'\}.$$

We now add group edges to form a clique in each group, and star edges to make the entire graph have diameter $2k$:

$$E_{group} = \{\{a,b\} : a,b \in A \cup B \wedge \Gamma^{-1}(a) = \Gamma^{-1}(b)\},$$
$$E_{star} = \{\{t, t_{y,k-1}\} : y \in V_R\} \cup \{\{t_{y,i}, t_{y,i+1}\} : y \in V_R \wedge i \in [k-2]\}.$$

This completes the reduction: the final edge set is

$$E_G = E_{in} \cup E_{con} \cup E_{path} \cup E_L \cup E_P \cup E_{so} \cup E_{sm} \cup E_M \cup E_{out} \cup E_{group} \cup E_{star}.$$

C.2 Analysis

C.2.1 Soundness

We will first show that the size of any $+k$-spanner of G is lower bounded by OPT_{MR} (the optimal solution of the Min-Rep instance $\widetilde{G}$). Let H be an arbitrary $+k$-spanner of G.

Definition C.1 Let $\{u,v\} \in E'$. A path between outer nodes $(u,i,k-1)$ and $(v,j,k-1)$ is called canonical if it has the form

$$(u,i,k-1) \to (u,i,k-2) \to \cdots \to (u,i,1) \to a \to b \to (v,j,1) \to (v,j,2) \to \cdots \to (v,j,k-1).$$

In other words, it goes through edges in E_{path}, then through a connection edge, then an inner edge, then a connection edges, and then more edges in E_{path}.

Note that any canonical path has length exactly $2k - 1$, and that $a \in \Gamma(u)$ and $b \in \Gamma(v)$. Since the distance in G from $(u, i, k - 1)$ and $(v, j, k - 1)$ is $k - 1$ via the path $(u, i, k - 1) \to m_{uv}^1 \to m_{uv}^2 \to \cdots \to m_{uv}^{k-2} \to (v, j, k - 1)$, a canonical path is a valid spanning path for this pair.

Definition C.2 *A $+k$-spanner H' of G is* canonical *if for all outer nodes $(u, i, k-1)$ and $(v, i, k-1)$ with $\{u, v\} \in E'$, there is a canonical path between $(u, i, k - 1)$ and $(v, i, k - 1)$*

Note that this definition of canonical uses the same value of i on both sides – this is to simplify the charging argument used in the proof of the next lemma.

Lemma C.3 *There is a canonical $+k$-spanner H' of G with $|E(H')| \leq 2k|E(H)|$.*

Proof: For each superedge $\{u, v\} \in E'$, let $a_{uv} \in \Gamma(u)$ and $b_{uv} \in \Gamma(v)$ be arbitrary inner nodes so that $\{a_{uv}, b_{uv}\} \in \widetilde{E}$, i.e. an pair of nodes which would cover the superedge. We create a new graph H' by starting with H, and then for each pair of outer nodes $(u, i, k - 1)$ and $(v, i, k - 1)$ where $\{u, v\} \in E'$ that do not have a canonical path, we add the canonical path between them through a_{uv} and b_{uv}. Slightly more formally, if any edges in the following path are missing, we add them (note that all of these edges are in G): $(u, i, k - 1) \to (u, i, k - 1) \to \cdots \to (u, i, 1) \to a_{uv} \to b_{uv} \to (v, i, 1) \to (v, i, 2) \to \cdots \to (v, i, k - 1)$.

Since H' includes H, it is clearly a $+k$-spanner. It is canonical by construction: if it is not canonical then there is a pair of outer nodes $(u, i, k - 1)$ and $(v, i, k - 1)$ with $\{u, v\} \in E'$ which is not spanned by a canonical path. But for every such pair we added a canonical path.

Thus it remains only to bound the size of H'. We do this by a charging argument: we will show how to charge edges we added to already existing edges of H, in such a way that no edge of H is charged more than $2k - 1$ times. This will clearly prove the lemma.

Suppose to get H' we added a canonical path to span $(u, i, k - 1)$ and $(v, i, k - 1)$. Note that this path has length exactly $2k - 1$. In order to charge this path to an edge of H, we first need to understand the way in which $(u, i, k - 1)$ and $(v, i, k - 1)$ could have been spanned in H. Since the distance between them in G is exactly $k - 1$ (using the path through M), in H there must be a path between them of length at most $2k - 1$, and this path must be non-canonical (or else we would not have added a canonical path). It is easy to verify that there are only three non-canonical paths between them of length at most $2k - 1$:

1. $(u, i, k - 1) \to m_{uv}^1 \to m_{uv}^2 \to \cdots \to m_{uv}^{k-2} \to (v, i, k - 1)$ (length $k - 1$), and

2. $(u, i, k - 1) \to \ell_{u,i,1} \to \ell_{u,i,2} \to \cdots \to \ell_{u,i,k-1} \to s_u \to m_{uv}^1 \to m_{uv}^2 \to \cdots \to m_{uv}^{k-2} \to (v, i, k - 1)$ (length $2k - 1$), and

3. $(u, i, k - 1) \to m_{uv}^1 \to m_{uv}^2 \to \cdots \to m_{uv}^{k-2} \to s_v \to \ell_{v,i,k-1} \to \ell_{v,i,k-2} \to \cdots \to \ell_{v,i,1} \to (v, i, k - 1)$ (length $2k - 1$).

Hence H must include at least one of these paths. All three of these paths include an outer edge: either $\{(u, i, k-1), m_{uv}^1\}$ or $\{(v, i, k-1), m_{uv}^{k-2}\}$ or both. We charge all edges on the canonical path we added to whichever of these outer edges exists in H (if they both exist in H, then we pick one arbitrarily). Clearly any two canonical paths will be charged to different outer edges, and thus each edge in H is charged at most $2k - 1$ times, proving the lemma. ∎

This now makes it easy to prove soundness.

Theorem C.4 *Any $+k$-spanner H of G has $|E(H)| \geq \frac{x}{2k} \cdot OPT_{MR}$*

Proof: By Lemma C.3, we just need to show that $|E(H')| \geq x \cdot OPT_{MR}$.

For $i \in [x]$, let $C_i \subseteq A \cup B$ be the set of inner nodes which in H' are adjacent to the associated outer node, i.e. $C_i = \{a \in A \cup B : \{(\Gamma^{-1}(a), i, 1), a\} \in E(H')\}$. We first claim that C_i is a REP-cover of $\tilde{G}$ for all $i \in [x]$. To see this, fix $i \in [x]$ and a superedge $\{u, v\} \in E'$. Since H' is canonical there is a canonical path $(u, i, k - 1) \to (u, i, k - 2) \to \cdots \to (u, i, 1) \to a \to b \to (v, i, 1) \to (v, i, 2) \to (\cdots \to (v, i, k - 1)$ in H'. Thus a and b are both in C_i, and since the middle edge of the canonical path is an edge in $\tilde{E}$ we know that C_i covers the superedge $\{u, v\}$. Hence C_i is a valid REP-cover for all $i \in [x]$, and thus $|C_i| \geq OPT_{MR}$

Now note that by definition, every node $a \in C_i$ is incident on a connection edge $\{a, (\Gamma^{-1}(a), i, 1)\}$. Since this edge is different for all $i \in [x]$ and $a \in C_i$, we immediately get that $|E(H')| \geq \sum_{i=1}^{x} |C_i| \geq x \cdot OPT_{MR}$. ∎

C.2.2 Completeness

We now show completeness: that there is a $+1$-spanner of G which does not cost too much *more* than OPT_{MR}. Let C be a REP-cover of size OPT_{MR}, and for each $u \in U \cup V$ let $C_u = C \cap \Gamma(u)$.

The subgraph H of G that we will analyze will include $E_{in} \cup E_{path} \cup E_L \cup E_P \cup E_{so} \cup E_{sm} \cup E_M \cup E_{group} \cup E_{star}$. In other words, the only edge sets we defined which it does not entirely include are E_{con} and E_{out}. Of these, we do not include any edges of E_{out}. We include connection edges to inner nodes in C, i.e. we also include all edges in the set $\{\{(u, i, 1), a\} : u \in U \cup V \wedge a \in C_u \wedge i \in [x]\}$.

We first analyze the size of H, and the later will prove that it is a $+k$-spanner of G. Recall that d is the degree of the supergraph G' (which is regular), and $|\Sigma|$ is the size of each group. Let $n' = |U \cup V|$.

Theorem C.5 *H has at most $x \cdot 7k^2 OPT_{MR} + 8k^2 n' d |\Sigma|^2$ edges*

Proof: The following size bounds are direct from the definitions:

$$|E_{in}| \leq n' d |\Sigma|^2, \qquad |E_{path}| \leq (k-2) n' x, \qquad |E_L| \leq k n' x,$$
$$|E_{so}| \leq n' x, \qquad |E_{sm}| \leq n' d, \qquad |E_M| \leq k n' d,$$
$$|E_{group}| \leq n' |\Sigma|^2, \qquad |E_{star}| \leq 3 n' k^2 x + 2 n' k (|\Sigma| + dk) \qquad |E_P| \leq k n' x$$

The number of connection edges we add is at most $x \cdot |C| = x \cdot OPT_{MR}$. Adding these all up, we get that

$$|E(H)| \leq x(OPT_{MR} + 7k^2 n') + 6k n' |\Sigma|^2 + 2n' k^2 d$$
$$\leq x(OPT_{MR} + 7k^2 n') + 8k^2 n' d |\Sigma|^2$$
$$\leq x \cdot 8k^2 OPT_{MR} + 8k^2 n' d |\Sigma|^2$$

where for the last inequality we used the fact that $OPT_{MR} \geq n'$. ∎

Note that this implies that if $x \geq d |\Sigma|^2$ then $|E(H)| \leq x \cdot 8k^2 OPT_{MR} + x \cdot 8k^2 OPT_{MR} \leq x \cdot 16k^2 OPT_{MR}$

Theorem C.6 *H is a $+k$-spanner of G.*

Proof: Because of E_{star}, the diameter of H is at most $2k$ and thus we only need to worry about spanning a pair of nodes if their distance in G is at most $k-1$.

We first consider the simple case of nodes at distance 1 (i.e. edges of G). The only edges of G missing from H are the outer edges and some connection edges. Consider a connection edge $\{(u,i,1),a\}$ which is not in H. Then since $C_u \neq \emptyset$ there is some $a' \in C_u$, and thus H has a path of length 2 spanning the missing edge: $(u,i,1) \to a' \to a$, where the first is a connection edge and the second is a group edge.

Now consider a missing outer edge $\{(u,i,k-1),m_{uv}^1\}$. In H there is a path of length $k+1$ between these nodes by using edges of E_L and E_{sm}, in particular the path $(u,i,k-1) \to \ell_{u,i,1} \to \ell_{u,i,2} \to \cdots \to \ell_{u,i,k-1} \to s_u \to m_{uv}^1$. A similar path exists for missing outer edges of the form $\{(v,i,k-1),m_{uv}^{k-2}\}$.

For the next case, consider two nodes at distance 2 in G. If the shortest path between them has zero or one edge in $E(G) \setminus E(H)$, then we know that their distance in H is at most $k+2$ as desired. The only pairs of nodes at distance 2 in G whose shortest path contains two edges from $E(G) \setminus E(H)$ are 1) outermost outer nodes corresponding to the same supernode, e.g. nodes $(u,i,k-1)$ and $(u,i',k-1)$ with $i \neq i'$, and 2) innermost outer nodes corresponding to the same supernode, e.g. nodes $(u,i,1)$ and $(u,i',1)$ with $i \neq i'$. The second case is simple: these nodes have a 2-path in H using a different shortest path of two connection edges to the same inner node in C_u. The first case is essentially why we added the P and Q nodes: there is a path of length either $k+1$ (if k is even) or $k+2$ (if k is odd) between $(u,i,k-1)$ and $(u,i',k-1)$ by using edges in E_P.

Now consider two nodes who are at distance more than 2 (but less than k) in G. If their shortest path contains at most 1 outer edge or connection edge, then the above discussion implies that they are spanned in H. Since they are at distance less than k, their shortest path cannot include *both* an outer edge and a connection edge. Hence we are only concerned with pairs whose shortest path contains more than one connection edge or more than one outer edge.

It is easy to verify that every shortest path of length less than k includes at most two connection edges, or at most two outer edges. Pairs that are connected by a shortest path with two connection edges are spanned in H by a path of length at most 2 longer than in G by routing around the missing connection edges.

The more interesting case is pairs of nodes that are connected by a shortest path which includes two outer edges. If these two outer edges directly follow each other on the shortest path, then we already showed that they are spanned by a path of length at most $k+2$. Hence using this detour only add at most $+k$ to the length of the shortest path and thus H has a path at most k longer than in G. On the other hand, if they do not directly follow each other on the shortest path, then the fact that the shortest path has length less than k implies that in fact the two nodes are outermost outer nodes $(u,i,k-1)$ and $(v,j,k-1)$ where $\{u,v\} \in E'$ (no other pairs have distance between 3 and $k-1$ in G with shortest path containing two non consecutive outer edges). By construction, there is a canonical path of length $2k-1$ in H between these nodes, and in G their distance is $k-1$, so H does span them.

Since this exhausts all cases, we have shown that H is a $+k$-spanner of G. ∎

C.2.3 Putting it Together

Theorem C.7 *For any constant $\epsilon > 0$ and any value $k \geq 3$ (not necessarily constant), there is no polynomial-time $2^{\log^{1-\epsilon} n}/k^3$-approximation algorithm for the $+k$-spanner problem unless $NP \subseteq DTIME(2^{\mathrm{polylog}(n)})$.*

Proof: We first instantiate x to $d|\Sigma|^2$. This means that the size of our reduction is

$$|V(H)| \leq k|V_R| \leq k(3n'kx + 2n'(|\Sigma| + dk)) \leq 3n'k^2 d|\Sigma|^2 + 2n'|\Sigma| + 2n'dk$$
$$\leq 7k^2 n' d|\Sigma|^2 \leq 7k^2 (n'|\Sigma|)^2.$$

Let OPT denote the size of the sparsest $+k$-spanner of H. Then Theorems C.4, C.5, and C.6 imply that

$$\frac{x}{2k} \cdot OPT_{MR} \leq OPT \leq x \cdot 16k^2 OPT_{MR}.$$

Putting the size and the approximation together (with the fact that the original Min-Rep instance had size $n'|\Sigma|$), we get that an $f(n)$-approximation for the $+k$-spanner problem would result in a $32k^3 \cdot f(7k^2 n^2)$-approximation for Min-Rep. Then the known hardness results for Min-Rep imply that this is at least $2^{\log^{1-\epsilon} n}$. Using a smaller value of ϵ in order to dominate the constant factors completes the proof. ∎

D The Proof of Height Reduction (Lemma 5.1)

In this section, we sketch the proof of the existence of the tree $\hat{T}_{\hat{r}}$ in Lemma 5.1. whose subtrees corresponding to an arborescence in the original graph with a bounded cost. In short, the tree $\hat{T}_{\hat{r}}$ in Lemma 5.1 in constructed by listing all paths rooted at r of length at most σ in the metric completion of G and then form a suffix tree on these paths. Our reduction is, indeed, the same as "path-splitting" technique in [13]. All we need is to show the cost guarantee using the famous Zelikovsky's Height Reduction Theorem.

Given a graph G, the *metric-completion* graph of G, denoted by $\Lambda(G)$, is a complete graph on the same vertex set as G, where each edge uv in $\Lambda(G)$ has weight equal to the distance of u, v in G. Thus, each edge uv of $\Lambda(G)$ is associated with a shortest u, v-path in G, denoted by $\phi(uv)$.

Theorem D.1 (Zelikovsky's Height Reduction Theorem [30, 25]) *For any arborescence J with edge weights $w : E(J) \to \mathbb{R}_0^+$, there exists an arborescence J' in the metric completion $\Lambda(J)$ of J with height σ and edge weights $w' : E(J') \to \mathbb{R}_0^+$ such that $\mathcal{L}(J) = \mathcal{L}(J')$ where $\mathcal{L}(J)$ and $\mathcal{L}(J')$ are leaves of J and J', respectively, and $w(J) = O(\sigma|\mathcal{L}(\mathcal{J})|^{1/\sigma})$.*

Proof: [Sketch of The Proof] We order vertices of J by depth-first-search order. Let $i = 0$, and let $V_0 = \mathcal{L}(J)$ be the set of "leaves" of J. We first add a copy of V_i to $\hat{J}$. Then we partition vertices of V_i into blocks $B_1, \ldots, B_q$, each of size Δ, according to the depth-first-search order. For each block B_j, we find the least common ancestor of vertices of B_j, denoted by a_j. We add edges from each $v \in B_j$ to a_j (which will be added to J'). Then we define $V_{i+1} = \{a_j\}_j$ (remove duplicate) and continue the same process on V_{i+1} until $|V_{i+1}| = 1$. At the termination, if the only vertex left in V_{i+1} is $v \neq r$, then we add a copy of r to J' and join v to r. We define the weight $w'(uv)$ of an edge $uv \in E(J')$ to be the weight of the unique u, v-path in J.

In each step, we pay a factor of $O(\Delta \cdot w(J))$ because paths from the same block may share edges, and they can also share edges with one block from the left and one from the right. Since there are only $\log_\Delta(|\mathcal{L}(J)|)$ steps, we pay a factor of $O(\Delta \cdot \log_\Delta(|\mathcal{L}(J)|))$. By adjusting the parameter Δ so that $\sigma = \log_\Delta(|\mathcal{L}(J)|)$ and thus $\Delta = |\mathcal{L}(J)|^{1/\sigma}$, we have the claimed upper bound. ∎

Building the tree $\hat{T}_{\hat{r}}$. Next, we construct the tree $\hat{T}_{\hat{r}}$ as in Lemma 5.1 by listing all paths of length (number of edges) at most σ in the metric completion of G (denoted by $\Lambda(G)$) that start from r and then form a suffix tree. Let $\mathbb{P}_\sigma^r$ denote the set of all paths in $\Lambda(G)$ that start at r and has length at most σ. We then define $\hat{T}_{\hat{r}} = (\hat{V}, \hat{E})$ as a suffix tree on the set $\mathbb{P}_\sigma^r$, i.e., the vertex and edge sets of $\hat{T}_r$ are

$$\hat{V} = \mathbb{P}_\sigma^r \quad \text{and} \quad \hat{E} = \{(Q, Qv) : Q, Qv \in \mathbb{P}_\sigma^r\}.$$

The mapping is defined to be the end vertex (resp., edge) of each path in $\mathbb{P}_\sigma^r$.

$$
\begin{aligned}
\Psi(Qv) &= v, & &\text{for all } Qv \in \hat{V} \text{ where } Q \in \mathbb{P}_\sigma^r, v \in V(G)\\
\Phi(Qu, Quv) &= \phi(uv), & &\text{for all } Qu, Quv \in \hat{V} \text{ where } Q \in \mathbb{P}_\sigma^r; u, v \in V(G)\\
\hat{w}_{\hat{u}\hat{v}} &= w(\phi(uv)), & &\text{for all } \hat{u}\hat{v} \in \hat{E} \text{ where } \psi(\hat{u}) = Qu \text{ and } \psi(\hat{v}) = Quv
\end{aligned}
$$

Now consider any arborescence J in G, and the arborescence J' of height σ as in Theorem D.1. It is easy to see that there is a tree $\hat{J}$ in $\hat{T}_{\hat{r}}$ such that $V(J') = \Psi(V(\hat{J}))$, and $E(J') = \Psi(E(\hat{J}))$. This simply follows by the construction of $\hat{T}$ that we list all the rooted-path of length at most σ in $\Lambda(G)$. For any edge $uv \in E(J')$, where $u = \psi(\hat{u})$ and $v = \psi(\hat{v})$, we know that $\hat{w}(\hat{u}\hat{v}) \leq w'(uv)$ because $\hat{w}(\hat{u}\hat{v})$ is the weight of a shortest u, v-path in G whereas $w'(uv)$ is the weight of a u, v-path in the subgraph J of G. This proves Lemma 5.1.