New Pairwise Spanners

Telikepalli Kavitha

Tata Institute of Fundamental Research, India
kavitha@tcs.tifr.res.in

Abstract

Let $G = (V, E)$ be an undirected unweighted graph on n vertices. A subgraph H of G is called an (all-pairs) purely additive spanner with stretch β if for every $(u, v) \in V \times V$, $\mathsf{dist}_H(u, v) \leq \mathsf{dist}_G(u, v) + \beta$. The problem of computing sparse spanners with small stretch β is well-studied. Here we consider the following relaxation: we are given $\mathcal{P} \subseteq V \times V$ and we seek a sparse subgraph H where $\mathsf{dist}_H(u, v) \leq \mathsf{dist}_G(u, v) + \beta$ for each $(u, v) \in \mathcal{P}$. Such a subgraph is called a pairwise spanner with additive stretch β and our goal is to construct such subgraphs that are sparser than all-pairs spanners with the same stretch. We show sparse pairwise spanners with additive stretch 4 and with additive stretch 6. We also consider the following special cases: $\mathcal{P} = S \times V$ and $\mathcal{P} = S \times T$, where $S \subseteq V$ and $T \subseteq V$, and show sparser pairwise spanners for these cases.

1998 ACM Subject Classification G.2.2 Graph Algorithms

Keywords and phrases undirected graphs, spanners, approximate distances, additive stretch

Digital Object Identifier 10.4230/LIPIcs.STACS.2015.513

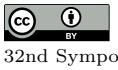
1 Introduction

Let $G = (V, E)$ be an undirected unweighted graph on n vertices. A subgraph H of G is called a spanner with multiplicative stretch α and additive stretch β if for every pair $(u, v) \in V \times V$, we have $\delta_H(u, v) \leq \alpha \cdot \delta_G(u, v) + \beta$, where $\delta_G(u, v)$ (similarly, $\delta_H(u, v)$) is the u-v distance in G (resp., H). The objective in spanner problems is to construct a subgraph H that is as sparse as possible, however for each pair of vertices (u, v), the u-v distance in H is close to the u-v distance in G, i.e., the stretch is small.

When the multiplicative stretch α is 1 and the additive stretch β is $O(1)$, the spanner H is said to be *purely additive*. We currently know purely additive spanners only for additive stretches 2, 4, and 6. The additive stretch 2 spanner has size $O(n^{3/2})$ [1], the additive stretch 4 spanner has size $\tilde{O}(n^{7/5})$ [10], and the additive stretch 6 spanner has size $O(n^{4/3})$ [6]. In this paper we consider the problem of obtaining sparser subgraphs for the following relaxed problem: distances for all pairs in $V \times V$ need not be well-approximated in the subgraph – here we have a subset $\mathcal{P} \subseteq V \times V$ of *critical* pairs and we seek a subgraph H where $\delta_H(u, v) \leq \delta_G(u, v) + O(1)$ for every $(u, v) \in \mathcal{P}$.

Our goal is to find a subgraph H that is sparser (for a large range of values of $|\mathcal{P}|$) than the all-pairs spanner with the same additive stretch. This problem was first studied by Coppersmith and Elkin [12] who sought subgraphs where distances for pairs in $\mathcal{P}$ were exactly preserved. They called such subgraphs *pairwise preservers* and showed such subgraphs of size $O(\min\{n\sqrt{|\mathcal{P}|}, n + |\mathcal{P}|\sqrt{n}\})$ for any $\mathcal{P} \subseteq V \times V$. They left it as an open question to study the approximate variants of pairwise preservers.

This question was studied by Cygan et al. [15] who called such subgraphs *pairwise spanners* or $\mathcal{P}$-spanners. A trade-off between the additive stretch and size of a $\mathcal{P}$-spanner was shown in [15], where the least stretch $\mathcal{P}$-spanner had additive stretch 4 with size $O(n^{4/3}|\mathcal{P}|^{1/6})$ and the sparsest $\mathcal{P}$-spanner had size $\tilde{O}(n|\mathcal{P}|^{1/4})$ with additive stretch $4 \log n$.



32nd Symposium on Theoretical Aspects of Computer Science (STACS 2015).
Editors: Ernst W. Mayr and Nicolas Ollinger; pp. 513–526

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany


Subsequently, a pairwise spanner of size $\tilde{O}(n|\mathcal{P}|^{1/3})$ and additive stretch 2 was shown in [20]. We show the following results here.

▶ **Theorem 1.** *Let $G = (V, E)$ be an undirected unweighted graph. Then for any $\mathcal{P} \subseteq V \times V$, the following subgraphs can be constructed in polynomial time (where $|V| = n$):*
1. *a $\mathcal{P}$-spanner of size $\tilde{O}(n \cdot |\mathcal{P}|^{2/7})$ and additive stretch 4,*
2. *a $\mathcal{P}$-spanner of size $O(n \cdot |\mathcal{P}|^{1/4})$ and additive stretch 6.*

Comparing our pairwise spanners with pairwise preservers and all-pairs spanners: the pairwise spanner in Theorem 1.1 is sparser than the $\tilde{O}(n^{7/5})$-sized all-pairs spanner with additive stretch 4 when $|\mathcal{P}|$ is $o(n^{7/5})$ and it is sparser than the $O(n+|\mathcal{P}|\sqrt{n})$-sized pairwise preserver when $|\mathcal{P}|$ is $\omega(n^{7/10})$. The pairwise spanner in Theorem 1.2 is sparser than the above pairwise preserver when $|\mathcal{P}|$ is $\omega(n^{2/3})$ and it is sparser than the $O(n^{4/3})$-sized all-pairs spanner with additive stretch 6 when $|\mathcal{P}|$ is $o(n^{4/3})$.

A natural setting for $\mathcal{P} \subseteq V \times V$ is when the set $\mathcal{P}$ of relevant pairs is $S \times T$, where S is one set of endpoints or *sources* and T is another set of endpoints or *destinations* and we seek a subgraph where s-t distances are well-approximated for every $(s, t) \in S \times T$. The case where $S = T$ was considered earlier and an $(S \times S)$-spanner of size $O(n\sqrt{|S|})$ with additive stretch 2 is known [15, 25]. Here we seek $(S \times T)$-spanners that are sparser than an $(S \cup T) \times (S \cup T)$-spanner or an $(S \times V)$-spanner (for instance, when $|S| \ll |T| \ll n$). To the best of our knowledge, $(S \times T)$-spanners are being studied for the first time here and we will refer to them as *ST-spanners*. We show the following result on ST-spanners.

▶ **Theorem 2.** *For any subsets S and T of V, an ST-spanner of size $O(n \cdot (|S|\,|T|)^{1/4})$ and additive stretch 4 can be constructed in polynomial time.*

Thus the above result combines the size of the pairwise spanner in Theorem 1.2 with the stretch of the one in Theorem 1.1, in other words, we get a $\mathcal{P}$-spanner of size $O(n|\mathcal{P}|^{1/4})$ with additive stretch 4 when $\mathcal{P} = S \times T$. Our next theorem shows that Theorem 1 can be further improved for *sourcewise spanners*, i.e., when $\mathcal{P} = S \times V$. Sourcewise spanners form a natural and interesting class of general $\mathcal{P}$-spanners and ST-spanners.

▶ **Theorem 3.** *The following subgraphs can be constructed in polynomial time for any $S \subseteq V$:*
1. *an $(S \times V)$-spanner of size $\tilde{O}(n \cdot (n|S|)^{2/9})$ and additive stretch 4,*
2. *an $(S \times V)$-spanner of size $O(n \cdot (n|S|)^{1/5})$ and additive stretch 6.*

Sourcewise spanners of size $\tilde{O}(n \cdot (n|S|)^{1/4})$ and additive stretch 2 were shown in [20]. Trade-off results between the size and additive stretch of sourcewise spanners were shown in [15, 21]. We show a size vs additive stretch trade-off for ST-spanners in Theorem 4.

▶ **Theorem 4.** *For any integer $k \geq 1$ and $S, T \subseteq V$, an ST-spanner with additive stretch $2k$ and size $\tilde{O}(n \cdot (|S|^{\gamma}|T|)^{1/(2\gamma+1)})$, where $\gamma = k+1$, can be constructed in polynomial time.*

By setting T to be the set of *cluster centers* (defined in Section 2) in Theorem 4, we get for $k \geq 2$, sourcewise spanners with additive stretch $2k$ and size $\tilde{O}(n^{1+1/r}|S|^{k/r})$, where $r = 2k + 2$. This matches the current best trade-off for sourcewise spanners (from [21]). An advantage with our construction is that it is deterministic, hence the bound is on the worst case size of the sourcewise spanner constructed here while the construction in [21] was randomized, hence the bound there was on the expected size of the sourcewise spanner.

An all-pairs purely additive spanner of size $O(n^{1+\frac{2\delta}{2\delta+1}})$ can be obtained from a $\mathcal{P}$-spanner with additive stretch $O(1)$ and size $O(n \cdot |\mathcal{P}|^{\delta})$ by taking $\mathcal{P} = T \times T$, where $T = \{\text{cluster centers}\}$; thus if $\delta < 1/4$, we get an all-pairs purely additive spanner of size $O(n^{1+\epsilon})$, where $\epsilon < 1/3$. No such purely additive spanners are currently known.

1.1 Background and Related Results

Graph spanners were introduced by Peleg and Schaffer [23] in 1989. The problem of efficiently constructing sparse spanners with a small multiplicative stretch is well-understood: Althöfer et al. [2] in 1993 showed that in any weighted graph G on n vertices and for any integer $k \geq 1$, there is a spanner of size $O(n^{1+1/k})$ with multiplicative stretch $2k-1$. More efficient constructions of spanners in weighted graphs can be found in [8, 27, 26]. There are several applications in graph/network algorithms that use spanners: for instance, approximate shortest paths [3, 11, 17], approximate distance oracles [28, 7, 5], labeling schemes [22, 19], network design [24], routing [4, 13, 14].

For unweighted graphs, we seek spanners with purely additive stretch – the first such spanner was by Aingworth et al. [1] (with followup work in [16, 18]) which showed a spanner with additive stretch 2 and size $O(n^{3/2})$. The additive stretch 6 spanner is due to Baswana et al. [6] and the additive stretch 4 spanner is due to Chechik [10]. The current best trade-off between sparsity and additive stretch is from [10]: for any $\delta \in [3/17, 1/3)$, there is a subgraph of size $\tilde{O}(n^{1+\delta})$ and additive stretch $\tilde{O}(n^{(1-3\delta)/2})$.

Bollobás et al. [9] were the first to study a variant of pairwise preservers – they studied *D-preservers* where the problem was to compute a sparse pairwise preserver for the set $\mathcal{P} = \{(u,v) : \delta_G(u,v) \geq D\}$. As mentioned earlier, Coppersmith and Elkin [12] studied the general problem of pairwise preservers for any $\mathcal{P} \subseteq V \times V$.

The study of pairwise spanners was initiated by Cygan et al. [15] who showed the following trade-off for pairwise spanners – for any $\mathcal{P} \subseteq V \times V$ and integer $k \geq 1$: additive stretch $4k$ and size $O(n^{1+1/r} \cdot (k|\mathcal{P}|)^{k/2r})$, where $r = 2k+1$ and the following trade-off between size and additive stretch for sourcewise spanners – for any subset S and any integer $k \geq 1$: additive stretch $2k$ and size $O(n^{1+1/r}(k|S|)^{k/r})$, where $r = 2k+1$. Parter [21] showed sparse *multiplicative* sourcewise spanners and a lower bound of $\Omega(n|S|^{1/k}/k)$ on the size of a sourcewise spanner with additive stretch $2(k-1)$, for any integer $k \geq 1$.

1.2 Techniques

All our algorithms start with the clustering step where vertices are grouped into *clusters* and at the end of this step, we are left with a post-clustering subgraph that contains all edges of G, except some inter-cluster edges. In each of our algorithms, the rest of the algorithm has to decide which of these missing inter-cluster edges should get added to the post-clustering subgraph. We use the steps of path-buying and path-hitting to make these decisions.

The *path-buying* step was introduced in [6] and has been used in several spanner algorithms [15, 20, 21]; here each shortest path is "evaluated" and if it is affordable, the path is bought, i.e., its missing edges are added to the current subgraph. Otherwise it will have to be the case that the path is already well-approximated in the current subgraph. We use path-buying with appropriate value functions in our algorithms for pairwise/sourcewise spanner with additive stretch 6 and ST-spanner with additive stretch 4.

We use *path-hitting* in our other algorithms. Path-hitting was first seen in the near-quadratic time algorithm in [29] for an $\tilde{O}(n^{4/3})$-sized spanner with additive stretch 6. This technique aims at hitting the neighborhood of each shortest path: this was done in [29] by randomly sampling vertices and appropriate near-shortest paths between the sampled vertices and other vertices were added to form the spanner. In our path-hitting subroutines here, we select a small number of clusters so that certain critical subpaths of all our relevant shortest paths are hit, i.e., for each such subpath, there will be some cluster among our selected ones that intersects it. Finally, our subgraph will contain shortest paths between

appropriate pairs of selected clusters.

Section 2 describes the clustering step and other preliminaries. Theorems 1 and 2 are shown in Section 3. Section 4 has our results on sourcewise spanners and Section 5 has the trade-off result for ST-spanners. Due to space constraints, the proofs of some lemmas are omitted; they will be included in the full version of the paper.

2 Preliminaries

A *clustering* of a graph $G = (V, E)$ is a collection $\{C_1, \ldots, C_r\}$ where each C_i is a subset of vertices and $C_i \cap C_j = \emptyset$ for $i \neq j$. Note that we do not require $\cup_i C_i$ to be equal to V; the vertices in $V - \cup_i C_i$ will be called *unclustered*. Associated with each cluster C_i is a vertex called its *center*, denoted by center(C_i), with the following property: every vertex in C_i is a neighbor of center(C_i). Note that center$(C_i) \notin C_i$.

Given a graph G and an integer h, where $1 \leq h \leq n$, the following simple procedure constructs a clustering $\mathcal{C}_h = \{C_1, \ldots, C_r\}$ and returns $\langle \mathcal{C}_h, U \rangle$, where $U = V - \cup_i C_i$ is the set of unclustered vertices.

– Initially all vertices are unclustered and $\mathcal{C}_h$ is empty. While there is a vertex v with at least h unclustered neighbors, we form a new cluster C by picking any h of these neighbors of v and these vertices are now clustered; v becomes center(C) and C gets added to $\mathcal{C}_h$. When no vertex has h or more unclustered neighbors, this procedure terminates and returns $\langle \mathcal{C}_h, U \rangle$, where U is the set of unclustered vertices.

It is easy to see that every pair of clusters is disjoint and each cluster C is a collection of h vertices. So $|\mathcal{C}_h|$, the number of clusters, is at most n/h. Associated with $\mathcal{C}_h$ is a post-clustering subgraph G_h that consists of all the edges incident to unclustered vertices, all *intra-cluster* edges (i.e., edges (u, v) where both u and v belong to the same cluster), as well as the edges (v, c), where v is a clustered vertex and c is the center of v's cluster. It can be shown that G_h has $O(nh)$ edges. We define the *cost* of a path below.

▶ **Definition 5.** For any path ρ in G, let $\mathsf{cost}(\rho)$ denote the number of edges of ρ that are missing in the post-clustering subgraph G_h.

The following lemma from [15] gives a lower bound on the number of distinct clusters that are incident on a shortest path ρ whose cost is t or more. We say a cluster C and a shortest path ρ intersect each other if C and ρ have at least one vertex in common.

▶ **Lemma 6** (from [15]). *Let ρ be a shortest path in G with $\mathsf{cost}(\rho) \geq t$. Then there are at least $t/2$ distinct clusters of $\mathcal{C}_h$ that intersect ρ.*

Another subroutine that our algorithms will use is that of efficiently computing a small-sized "hitting set" $\mathcal{A} \subseteq \mathcal{C}_h$ for a set Q of paths. We define such a set $\mathcal{A}$ below.

▶ **Definition 7.** Let Q be a set of paths. Then $\mathcal{A} \subseteq \mathcal{C}_h$ is a *hitting set* for Q if for every path $q \in Q$, there is at least one cluster in $\mathcal{A}$ that intersects q.

When each $q \in Q$ has several clusters incident on it, a small-sized hitting set can be computed easily by the greedy algorithm. We know from Lemma 6 that if a shortest path has many missing edges in G_h, then it has several distinct clusters incident on it. Lemma 8 uses this property to obtain a small-sized hitting set for such a set Q of paths.

▶ **Lemma 8.** *Let Q be a set of shortest paths such that each $q \in Q$ has at least λ missing edges in G_h. Then the greedy algorithm finds a hitting set $\mathcal{A} \subseteq \mathcal{C}_h$ of size $O(n/(h\lambda) \cdot \log |Q|)$ for Q.*

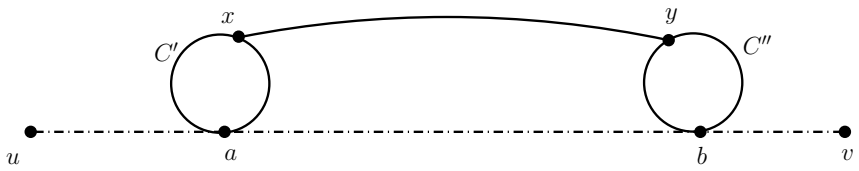


Figure 1 If $\delta_H(a,b) > \delta_G(a,b)+4$ before Step 5 of our algorithm, then we would have bought a path $SP(x,y)$ of length at most $\delta_G(a,b)$ between C' and C'' in Step 5. The u-v path $u{-}a{-}x{-}y{-}b{-}v$ has length at most $\delta_G(u,v)+4$.

It will be convenient to assume that there is a *unique* shortest path between any pair of vertices. So for every pair of vertices u and v, we will choose one u-v shortest path in G as *the* u-v shortest path (denoted by $SP(u,v)$) and we will ensure that every subpath of a chosen shortest path is again a chosen shortest path. Let $\mathcal{F} = \{\rho_1, \ldots, \rho_{\binom{n}{2}}\}$ be the set of all these chosen shortest paths in G.

Let $Q \subseteq \mathcal{F}$ be the set of shortest paths p such that $\mathsf{cost}(p) \geq (n \log n)/h^2$. Lemma 8 tells us that Q has a hitting set $\mathcal{X}_h$ of size $O(h)$. Consider the step of augmenting the post-clustering subgraph G_h with the edges of BFS trees rooted at the centers of clusters in $\mathcal{X}_h$. As there are only $O(h)$ such trees, the total number of edges added by this augmentation is $O(nh)$. The following lemma from [20] is a useful consequence of this augmentation.

▶ **Lemma 9.** *Let u and v be a pair of vertices such that $\mathsf{cost}(SP(u,v)) \geq (n \log n)/h^2$. Then the augmented post-clustering graph has a u-v path of length at most $\delta_G(u,v)+2$.*

3 Pairwise Spanners

In this section we prove Theorems 1 and 2 stated in Section 1. We first describe the construction of a $\mathcal{P}$-spanner of size $\tilde{O}(n|\mathcal{P}|^{2/7})$ with additive stretch 4. The input is an undirected unweighted graph $G=(V,E)$ along with $\mathcal{P} \subseteq V \times V$.

Our algorithm starts by running the clustering step with an appropriate parameter h and augments the post-clustering graph G_h by adding BFS trees rooted at the centers of clusters in $\mathcal{X}_h$, where $\mathcal{X}_h$ is an $O(h)$-sized hitting set for $Q = \{\rho \in \mathcal{F} : \mathsf{cost}(\rho) \geq (n \log n)/h^2\}$. Call the resulting graph H.

We are ready to buy 2ℓ new edges for each $(u,v) \in \mathcal{P}$, for an appropriate parameter ℓ. If $p = SP(u,v)$ is *expensive*, i.e., $\mathsf{cost}(p) > 2\ell$, then we buy only a prefix p' and a suffix p'' of p such that $\mathsf{cost}(p') = \mathsf{cost}(p'') = \ell$. The main idea here is *path-hitting* – we find small-sized sets $\mathcal{A}$ and $\mathcal{B}$ of clusters so that there is a cluster $C' \in \mathcal{A}$ that intersects p' and there is a cluster in $C'' \in \mathcal{B}$ that intersects p''.

For each pair of clusters $(C_0, C_1) \in \mathcal{A} \times \mathcal{B}$, we add to H at most one shortest path over all pairs of vertices in $C_0 \times C_1$. Then we have an approximate shortest path between u and v of the form $u - C' \rightsquigarrow C'' - v$, where $C' \in \mathcal{A}$ and $C'' \in \mathcal{B}$, and we will show that this resulting u-v path has additive stretch 4 (see Figure 1).

Our algorithm is presented below, let $h = \lceil |\mathcal{P}|^{2/7} \log^{3/7} n \rceil$ and $\ell = \lceil (n \log^{3/2} n)/h^{5/2} \rceil$.

1. Run the clustering step with the above h and let G_h be the post-clustering subgraph. Augment G_h with $O(h)$ BFS trees as described above. Let H be the resulting graph.
2. For each $(u,v) \in \mathcal{P}$ do: (let $\rho = SP(u,v)$)
   (a) if $\mathsf{cost}(\rho) \leq 2\ell$ then add all the missing edges of ρ to H.

(b) else add to H all the missing edges of $\mathsf{prefix}(\rho)$ and $\mathsf{suffix}(\rho)$, where $\mathsf{prefix}(\rho)$ (similarly, $\mathsf{suffix}(\rho)$) is the minimal prefix (resp., suffix) of ρ that has ℓ edges missing in G_h.

3. For each $p \in \mathcal{F}$ such that $\mathsf{cost}(p) > 2\ell$ do: {*recall that $\mathcal{F}$ is the set of all shortest paths*}
   - let $Q_0 = \{\mathsf{prefix}(p) : p \in \mathcal{F}\}$ and $Q_1 = \{\mathsf{suffix}(p) : p \in \mathcal{F}\}$.
4. Determine $\mathcal{A}$ and $\mathcal{B}$ greedily, where $\mathcal{A}$ = hitting set for Q_0 and $\mathcal{B}$ = hitting set for Q_1.
5. For each $C_0 \in \mathcal{A}$ and $C_1 \in \mathcal{B}$ do:
   - if there is a pair $(x, y) \in C_0 \times C_1$ such that $\delta_H(x, y) > \delta_G(x, y) + 4$, then find such a pair $(x', y') \in C_0 \times C_1$ with least $\delta_G(x', y')$ and add to H all the missing edges in $SP(x', y')$.

Lemma 10 shows that H is a $\mathcal{P}$-spanner with additive stretch 4 and Lemma 11 bounds the size of the subgraph H. Thus Theorem 1.1 follows.

▶ **Lemma 10.** *Let H be the subgraph computed by the algorithm above. Then $\delta_H(u, v) \leq \delta_G(u, v) + 4$ for each $(u, v) \in \mathcal{P}$.*

Proof. Consider any pair $(u, v) \in \mathcal{P}$ and let $\rho = SP(u, v)$. If $\mathsf{cost}(\rho) \leq 2\ell$ then by Step 2(a), the entire path ρ is present in H and so $\delta_H(u, v) = \delta_G(u, v)$ in this case. If $\mathsf{cost}(\rho) > 2\ell$ then there is a cluster $C' \in \mathcal{A}$ incident on the prefix ρ' of ρ and a cluster $C'' \in \mathcal{B}$ incident on the suffix ρ'' of ρ.

Let a be the first vertex of C' incident on ρ and b be the last vertex of C'' incident on ρ (see Figure 1). If $\delta_H(a, b) > \delta_G(a, b) + 4$ before Step 5 of our algorithm, then we would have bought a path of length at most $\delta_G(a, b)$ between C' and C'' in Step 5. Thus after Step 5, we have $\delta_H(a, b) \leq \text{diameter}_H(C') + \delta_H(C', C'') + \text{diameter}_H(C'') \leq \delta_G(a, b) + 4$.

In Step 2(b) of our algorithm, we would have added all edges in the u-a subpath and the b-v subpath of ρ, so $\delta_H(u, a) = \delta_G(u, a)$ and $\delta_H(b, v) = \delta_G(b, v)$. Thus at the end of the algorithm, we have $\delta_H(u, v) \leq \delta_H(u, a) + \delta_H(a, b) + \delta_H(b, v) \leq \delta_G(u, v) + 4$. ◀

▶ **Lemma 11.** *The size of the final subgraph H is $O(nh)$, where $h = \lceil |\mathcal{P}|^{2/7} \log^{3/7} n \rceil$.*

Proof. Initially the size of H is the size of the post-clustering graph G_h, which is $O(nh)$. The size of H remains $O(nh)$ after the augmentation with $O(h)$ BFS trees. We buy at most 2ℓ edges per pair in $\mathcal{P}$ in Step 2, so this adds to $2\ell|\mathcal{P}|$. For each $(C_0, C_1) \in \mathcal{A} \times \mathcal{B}$, we buy at most one shortest path – also such a shortest path has at most $(n \log n)/h^2$ missing edges in G_h; otherwise it would already have been approximated within an additive stretch of 2 in H by some BFS tree (by Lemma 9). So we buy at most $(n \log n)/h^2$ edges per $(C_0, C_1) \in \mathcal{A} \times \mathcal{B}$.

It follows from Lemma 8 that $|\mathcal{A}|$ and $|\mathcal{B}|$ are at most $O(n \log n/(h\ell))$. Thus the total number of edges bought in Step 5 is $O((n \log n)/h^2 \cdot (n \log n/h\ell)^2)$. This is $O(nh)$ since $\ell = \lceil (n \log^{3/2} n)/h^{5/2} \rceil$. Thus the total size of H is $O(nh + \ell|\mathcal{P}|)$. Substituting the values of ℓ and h, it follows that the size of H is $O(n\,|\mathcal{P}|^{2/7} \log^{3/7} n)$. ◀

A pairwise spanner with additive stretch 6. We now show a simple algorithm to construct a $\mathcal{P}$-spanner of size $O(n|\mathcal{P}|^{1/4})$ and additive stretch 6. The main step here is *path-buying* – for each shortest path ρ whose endpoints are in $\mathcal{P}$, if ρ is "affordable", then we *buy* ρ, i.e., add to H all the missing edges of ρ.

Definition 12 captures the value of a path. For any pair of clusters C_1, C_2 in a subgraph H, let $\delta_H(C_1, C_2) = \min\{\delta_H(a, b) : a \in C_1, b \in C_2\}$. Similarly, for any pair of clusters C_1, C_2 incident on a path p, let $\delta_p(C_1, C_2)$ be the minimum value of $\delta_p(x, y)$, where $x \in C_1$ and $y \in C_2$ lie on path p.

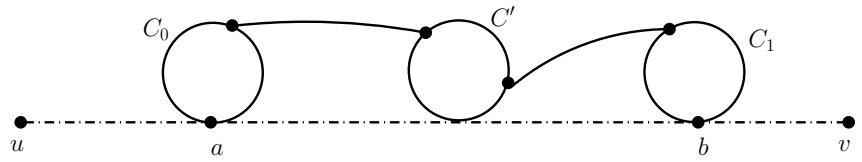


Figure 2 We have $\delta_H(C_0, C') \leq \delta_\rho(C_0, C')$ and $\delta_H(C', C_1) \leq \delta_\rho(C', C_1)$. Since the diameter of each cluster is 2, this implies $\delta_H(a, b) \leq \delta_G(a, b) + 6$.

▶ **Definition 12.** For any subgraph H of G and path p in G, let $\mathsf{value}_H(p)$ be the number of pairs of clusters (C_1, C_2) incident on p such that $\delta_p(C_1, C_2) < \delta_H(C_1, C_2)$.

Thus $\mathsf{value}_H(p)$ is the number of pairs of clusters incident on p whose distance in H would improve upon adding p to H. We also extend the cost function used in our earlier algorithm to $\mathsf{cost}_H()$: for any path p, let $\mathsf{cost}_H(p)$ be the number of edges of p that are missing in H. Our algorithm is described below. Let $h = \lceil |\mathcal{P}|^{1/4} \rceil$ and $\ell = \lceil n/h^3 \rceil$.

1. Run the clustering step with parameter h. Let H be the post-clustering subgraph G_h.
2. For each $(u, v) \in \mathcal{P}$ do: (let $\rho = SP(u, v)$)
   - **(a)** if $\mathsf{cost}_H(\rho) \leq \max\{4\ell,\ 8\,\mathsf{value}_H(\rho)/\ell\}$ then buy ρ.
   - **(b)** else buy $\mathsf{prefix}(\rho)$ and $\mathsf{suffix}(\rho)$.
     [$\mathsf{prefix}(\rho)$ is the minimal prefix of ρ such that $\mathsf{cost}_H(\mathsf{prefix}(\rho)) = \ell$ and similarly, $\mathsf{suffix}(\rho)$ is the minimal suffix of ρ such that $\mathsf{cost}_H(\mathsf{suffix}(\rho)) = \ell$.]

▶ **Lemma 13.** *For every pair $(u, v) \in \mathcal{P}$, we have $\delta_H(u, v) \leq \delta_G(u, v) + 6$.*

Proof. Let $\rho \in \mathcal{F}$ be the u-v shortest path, where $(u, v) \in \mathcal{P}$. This path ρ gets considered in Step 2 of our algorithm. If we buy ρ in Step 2(a), then $\delta_H(u, v) = \delta_G(u, v)$. Hence assume ρ was not bought, let $\mathsf{cost}_H(\rho) = t$ when ρ gets considered in Step 2. That is, at the time of processing ρ in Step 2, there are t edges of ρ missing in the current subgraph H.

We know that $t > 4\ell$, otherwise ρ would have been bought in Step 2(a). In Step 2(b), the prefix and suffix of ρ with ℓ edges missing in H, i.e., $\rho' = \mathsf{prefix}(\rho)$ and $\rho'' = \mathsf{suffix}(\rho)$, are bought. Let $q = \rho - (\rho' \cup \rho'')$, this is the *middle portion* of ρ consisting of $t - 2\ell$ missing edges. It follows from Lemma 6 that there are at least $\ell/2$ clusters incident on ρ' and on ρ''.

Let $\mathcal{A}_0$ (similarly, $\mathcal{A}_1$) be the set of clusters incident on ρ' (resp., ρ''). The subpath q has at least $(t - 2\ell)/2$ incident clusters (call this set of clusters $\mathcal{B}$). We will show Claim 1.

▶ Claim 1. There are clusters $C_0 \in \mathcal{A}_0$, $C' \in \mathcal{B}$, and $C_1 \in \mathcal{A}_1$ such that $\delta_H(C_0, C') \leq \delta_\rho(C_0, C')$ and $\delta_H(C', C_1) \leq \delta_\rho(C', C_1)$.

We will now assume the above claim and finish the proof of Lemma 13. Then we will prove Claim 1. This claim guarantees that the graph H contains a path $C_0 \rightsquigarrow C' \rightsquigarrow C_1$ such that the $C_0 \rightsquigarrow C'$ subpath has length at most $\delta_\rho(C_0, C')$ and the $C' \rightsquigarrow C_1$ subpath has length at most $\delta_\rho(C', C_1)$ (see Figure 2). Thus there are vertices $a \in \rho' \cap C_0$ and $b \in \rho'' \cap C_1$ such that $\delta_H(a, b) \leq \delta_G(a, b) + 6$ (via the $C_0 \rightsquigarrow C' \rightsquigarrow C_1$ path in H and $\mathrm{diameter}_H(C) \leq 2$ for all $C \in \mathcal{C}_h$). We buy all the missing edges in ρ' and in ρ'' in Step 2(b), hence we have $\delta_H(u, a) = \delta_G(u, a)$ and $\delta_H(b, v) = \delta_G(b, v)$. Since $\delta_H(u, v) \leq \delta_H(u, a) + \delta_H(a, b) + \delta_H(b, v)$, we get $\delta_H(u, v) \leq \delta_G(u, v) + 6$. ◀

Proof of Claim 1. Suppose there are no such clusters C_0, C', and C_1. Then for each cluster $C' \in \mathcal{B}$, either $\delta_\rho(C_0, C') < \delta_H(C_0, C')$ for every $C_0 \in \mathcal{A}_0$ or $\delta_\rho(C', C_1) < \delta_H(C', C_1)$

for every $C_1 \in \mathcal{A}_1$. Since $|\mathcal{A}_0| \geq \ell/2$, $|\mathcal{A}_1| \geq \ell/2$, and $|\mathcal{B}| \geq (t-2\ell)/2$, this means that

$$\mathsf{value}_H(\rho) \;\geq\; \frac{\ell}{2} \cdot \frac{t-2\ell}{2} \;>\; \frac{\ell}{2} \cdot \frac{t}{4} \;=\; \frac{\ell \cdot t}{8}, \tag{1}$$

where the second inequality follows from the fact that $t > 4\ell$, so $\ell < t/4$, thus $t - 2\ell > t/2$. Since we did not buy ρ in Step 2(a), it must be the case that $t > 8\,\mathsf{value}_H(p)/\ell$, i.e, $\mathsf{value}_H(\rho) < \ell \cdot t/8$. This contradicts Inequality (1). ◀

▶ **Lemma 14.** *The size of the final subgraph H is $O(n|\mathcal{P}|^{1/4})$.*

Proof. Initially the size of H is $O(nh)$ which is $O(n|\mathcal{P}|^{1/4})$. The total number of edges added in Step 2(b) is at most $2\ell|\mathcal{P}|$ since we buy at most 2ℓ edges per element in $\mathcal{P}$. Since $\ell = \lceil n/h^3 \rceil$ where $h = \lceil |\mathcal{P}|^{1/4} \rceil$, it follows that $O(\ell|\mathcal{P}|)$ is $O(n|\mathcal{P}|^{1/4})$. The total number of edges added in Step 2(a) is $\sum_\rho \mathsf{cost}(\rho)$ where the sum is over all the paths ρ that got bought in this step during the entire algorithm.

Let us evaluate $\sum_\rho \mathsf{cost}(\rho)$ for the paths ρ bought in Step 2(a) – this is at most $4\ell|\mathcal{P}| + \sum_\rho 8\,\mathsf{value}_H(\rho)/\ell$ since we buy ρ only when $\mathsf{cost}_H(\rho) \leq \max\{4\ell,\ 8\,\mathsf{value}_H(\rho)/\ell\}$. Let us bound $\sum_\rho \mathsf{value}_H(\rho)/\ell$ where the sum is over all the paths bought.

We say a pair of clusters $(C_1, C_2) \in \mathcal{C}_h \times \mathcal{C}_h$ *supports* ρ if (C_1, C_2) contributes positively to $\mathsf{value}_H(\rho)$. Consider all the shortest paths that were supported by a fixed pair (C_1, C_2). We claim at most 5 of them could have been bought in our algorithm. This is because once a shortest path ρ supported by (C_1, C_2) gets bought, we have $\delta_H(C_1, C_2) \leq \delta_G(C_1, C_2) + 4$ since this path ρ is $SP(x, y)$ for some $(x, y) \in C_1 \times C_2$. Thereafter, every time a path supported by (C_1, C_2) gets bought, $\delta_H(C_1, C_2)$ (strictly) decreases. Thus

$$\sum_\rho \frac{\mathsf{value}_H(\rho)}{\ell} = \frac{1}{\ell} \sum_{(C_1,C_2)} \text{number of paths supported by } (C_1, C_2) \text{ that got bought} \leq \frac{5\,|\mathcal{C}_h|^2}{\ell},$$

where the middle sum is over all pairs of clusters $(C_1, C_2) \in \mathcal{C}_h \times \mathcal{C}_h$. Substituting $|\mathcal{C}_h| \leq n/h$ and $\ell = \lceil n/h^3 \rceil$, the right side above is bounded by $O(nh)$. Hence the size of H is $O(n|\mathcal{P}|^{1/4})$. ◀

Theorem 1.2 follows from Lemmas 13 and 14. This finishes the proof of Theorem 1 stated in Section 1.

An ST-spanner with additive stretch 4. The input here is $G = (V, E)$ along with $S \subseteq V$ and $T \subseteq V$. We assume without loss of generality that $|S| \leq |T|$. Our algorithm here again uses *path-buying*, however with a new value function that is defined below.

▶ **Definition 15.** For any subgraph H of G and any path p in G with one endpoint in S (call this vertex s) and the other endpoint in T (call this vertex t), define $\mathsf{value}_H(p)$ as follows:

$$\mathsf{value}_H(p) \;=\; \ell \cdot |\{C_0 : \delta_p(s, C_0) < \delta_H(s, C_0)\}| \;+\; 2 \cdot |\{(C_1, C_2) : \delta_p(C_1, C_2) < \delta_H(C_1, C_2)\}|,$$

where all the clusters C_0, C_1, C_2 above have to be incident on p and the value $\ell = \lceil n/h^3 \rceil$ (the parameter h will be set to $\lceil (|S|\,|T|)^{1/4} \rceil$).

In other words, $\mathsf{value}_H(p) = \ell\alpha_1 + 2\alpha_2$, where α_1 is the number of clusters incident on p whose distance to s via p is better than the current distance to s in H and α_2 is the number of pairs of clusters incident on p whose distance in p is better than their current distance in H. Our algorithm is described below. Let $h = \lceil (|S|\,|T|)^{1/4} \rceil$.

1. Run the clustering step with parameter h. Let H be the post-clustering subgraph G_h.
2. For each $(s,t) \in S \times T$ do: (let $p = SP(s,t)$)
   - **(a)** If $\mathsf{cost}_H(p) \leq \max\{2\ell,\ 4\,\mathsf{value}_H(p)/\ell\}$ then add to H all the missing edges of p.
   - **(b)** Else buy the minimal suffix p' of p such that $\mathsf{cost}_H(p') = \ell$, i.e., add to H the last ℓ missing edges of p.

Lemma 16 states the correctness of the above algorithm. Its proof (which is omitted here) is similar to the proofs of Lemmas 13 and 14. Theorem 2 stated in Section 1 thus follows.

▶ **Lemma 16.** *The size of the final subgraph H is $O(n \cdot (|S|\,|T|)^{1/4})$. For every pair $(s,t) \in S \times T$, we have $\delta_H(s,t) \leq \delta_G(s,t) + 4$.*

4 Sourcewise spanners

The input here is $G = (V,E)$ along with a subset of V, which is the set of *sources*. In this section we will prove Theorem 3. We first show Theorem 3.2 whose proof follows quite easily from our ST-spanner with additive stretch 4 (shown in the previous section). Let $S' \subseteq V$ be the set of sources here – we will be using the symbols S and T for the 2 sets in the ST-spanner algorithm, so we use the symbol S' to refer to the set of sources here. Broadly speaking, we take the sets S and T in the ST-spanner to be the set S' and the set of all cluster centers to get a sourcewise spanner.

However we need to be careful about which of S,T becomes S' and which becomes the set of cluster centers: recall that our algorithm for ST-spanners assumed $|S| \leq |T|$. So we assign the sets S and T as follows: if $|S'| \leq n^{2/3}$ then assign $S = S'$ else assign $T = S'$. More precisely, our algorithm does the following:

1. Run the clustering step with parameter $h = \lceil (n|S'|)^{1/5} \rceil$. Let H be the post-clustering subgraph G_h. Let $T' = \{\text{cluster centers}\}$.
2. If $|S'| \leq n^{2/3}$ then set $S = S'$ and $T = T'$; else set $S = T'$ and $T = S'$.
3. Run Step 2 of our ST-spanner algorithm with additive stretch 4 (the parameter $\ell = \lceil n/h^3 \rceil$); return the ST-spanner H.

It can be shown (proof omitted here) that the subgraph returned by the ST-spanner algorithm is an $(S' \times V)$-spanner with additive stretch 6 and that this subgraph has size $O(n \cdot (n|S'|)^{1/5})$. Thus Theorem 3.2 stated in Section 1 follows.

We now prove Theorem 3.1 by describing an algorithm to construct a sparse $(S \times V)$-spanner with additive stretch 4, where $S \subseteq V$ is the set of sources. We run the clustering step with $h = \lceil (|S| n \log^2 n)^{2/9} \rceil$ and initialize the subgraph H to the post-clustering subgraph G_h. Then we augment H with BFS trees rooted at $O(h)$ selected cluster centers so that every path $p \in \mathcal{F}$ with $\mathsf{cost}(p) \geq (n \log n)/h^2$ has some adjacent cluster center that has been selected. For every $s \in S$ and cluster C, we pick the shortest path between s and its closest vertex $x \in C$ (in case of ties, x is chosen arbitrarily) and call it $SP(s,C)$ henceforth.

Let $\mathcal{R} \subseteq \mathcal{F}$ be the collection of paths $SP(s,C)$ where $s \in S$ and $C \in \mathcal{C}_h$. Corresponding to each $\rho \in \mathcal{R}$, we are ready to buy λ new edges, where $\lambda^2 = (n \log n)^2/h^5$. So if $\mathsf{cost}(\rho) \leq \lambda$, then we buy ρ. For any $\rho \in \mathcal{R}$, if $\mathsf{cost}(\rho) > \lambda$, then let ρ_0 to be the minimal suffix of ρ that has $\lfloor \lambda \rfloor$ missing edges in the post-clustering subgraph G_h; let Q_0 be the set containing all such suffixes ρ_0. We now use *path-hitting* to determine $\mathcal{A}_0 \subseteq \mathcal{C}_h$ so that for each $\rho_0 \in Q_0$ there is at least one cluster $C_0 \in \mathcal{A}_0$ that intersects ρ_0.

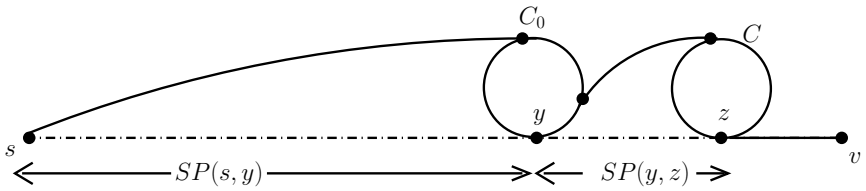


Figure 3 There are at most λ^2 missing edges in the path $SP(s,y)$ and at most λ missing edges in the path $SP(y,z)$. We will buy a path $s \rightsquigarrow C_0 \rightsquigarrow C$, where C is the last cluster on $SP(s,v)$.

1. For each $s \in S$ and $C_0 \in \mathcal{A}_0$: if there exists any $x \in C_0$ such that $\mathsf{cost}(SP(s,x)) \leq \lambda^2$ then buy $SP(s,x')$ where $x' \in C_0$ is the closest vertex to s that satisfies $\mathsf{cost}(SP(s,x')) \leq \lambda^2$ and also run the following step.

1.1. For each $C \in \mathcal{C}_h$ such that C_0 intersects $SP(s,u)$ for some $u \in C$ do:
- if there is some path $SP(a,b)$ where $(a,b) \in C_0 \times C$ with $\mathsf{cost}(SP(a,b)) \leq \lambda$ and buying this path improves $\delta_H(s,C)$, then buy $SP(a,b)$.

We now deal with $\rho \in \mathcal{R}$ such that $\mathsf{cost}(\rho) > \lambda^2$. For any such ρ, define ρ_1 be the minimal prefix of ρ with $\lfloor \lambda^2 \rfloor$ missing edges in the current H; let Q_1 be the set containing all such prefixes ρ_1. We again use path-hitting to determine $\mathcal{A}_1 \subseteq \mathcal{C}_h$ so that for each $\rho_1 \in Q_1$ there is at least one cluster $C_1 \in \mathcal{A}_1$ that intersects ρ_1. We run the following two steps now.

2. For each $(s, C_1) \in S \times \mathcal{A}_1$: if there exists any $w \in C_1$ such that $\mathsf{cost}(SP(s,w)) \leq \lambda^2$ then buy $SP(s,w')$ where $w' \in C_1$ is the closest vertex to s that satisfies $\mathsf{cost}(SP(s,w')) \leq \lambda^2$.

3. For each $(C_1, C) \in \mathcal{A}_1 \times \mathcal{C}_h$: if there is a pair $(a,b) \in C_1 \times C$ such that $\delta_H(a,b) > \delta_G(a,b) + 2$ and buying $SP(a,b)$ improves $\delta_H(C_1, C)$, then buy $SP(a,b)$.

▶ **Lemma 17.** *For every pair $(s,v) \in S \times V$, we have $\delta_H(s,v) \leq \delta_G(s,v) + 4$.*

Proof. Consider any pair $(s,v) \in S \times V$ and let $p = SP(s,v)$. We can assume that $\mathsf{cost}(p) < (n \log n)/h^2$, otherwise $\delta_H(s,v) \leq \delta_G(s,v) + 2$ due to adding certain BFS trees to H.

Let z be the last clustered vertex on p, i.e., z is the clustered vertex that is closest to v on this path p. Let C be the cluster containing z. Consider $\rho = SP(s,C)$. If $\mathsf{cost}(\rho) \leq \lambda$, then we would have bought ρ, which means $\delta_H(s,z) \leq \delta_G(s,z) + 2$, thus $\delta_H(s,v) \leq \delta_G(s,v) + 2$. So let us assume that $\mathsf{cost}(\rho) > \lambda$. We have two cases here:

Case(1): Suppose $\lambda < \mathsf{cost}(\rho) \leq \lambda^2$. There is a cluster $C_0 \in \mathcal{A}_0$ incident on the suffix ρ_0 of ρ. In other words, the path ρ restricted to the subpath $C_0 \rightsquigarrow C$ has at most λ missing edges. Let y be the first vertex of C_0 on ρ. Since $\mathsf{cost}(\rho) \leq \lambda^2$, we have $\mathsf{cost}(SP(s,y)) \leq \lambda^2$. So in Step (1) we either buy all the missing edges of $SP(s,y)$ or we already have a path $s \rightsquigarrow C_0$ in H of length at most $\delta_G(s,y)$. Thus we have $\delta_H(s,C_0) \leq \delta_G(s,y)$.

Now consider the y-z subpath of ρ (see Figure 3). We know that there are at most λ edges of $SP(y,z)$ that are missing in the subgraph H. Buying these λ edges causes $\delta_H(s,C) \leq \delta_H(s,C_0) + 2 + |SP(y,z)| \leq \delta_G(s,z) + 2$.

So either $\delta_H(s,C)$ is already at most $\delta_G(s,z)+2$ or we buy the missing edges of $SP(y,z)$ in Step (1.1) and make $\delta_H(s,C) \leq \delta_G(s,z) + 2$. Thus after this step, we have $\delta_H(s,v) \leq \delta_G(s,v) + 4$ using the path $s \rightsquigarrow C_0 \rightsquigarrow C \rightsquigarrow v$.

Case(2): We are left with the case when $\mathsf{cost}(\rho) > \lambda^2$. In this case we would have a cluster $C_1 \in \mathcal{A}_1$ incident on the prefix ρ_1 of ρ with at most λ^2 missing edges. Let x be the first vertex of C_1 on ρ (see Figure 4). While considering the pair (s, C_1), we would have either bought all the missing edges of $SP(s,x)$ in Step (2) (as there are at most λ^2 missing edges here) or we already have a path $s \rightsquigarrow C_1$ of length at most $\delta_G(s,x)$.

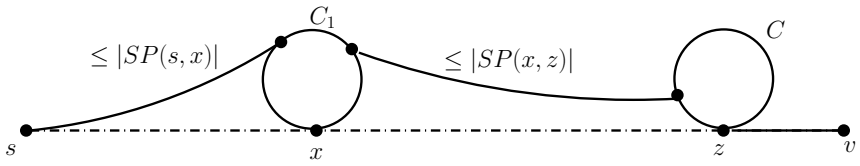


Figure 4 H has an $s \rightsquigarrow C_1$ path of length $|SP(s, C_1)|$. There is also a $C_1 \rightsquigarrow C$ path of length $|SP(x, z)|$.

Consider the pair $(C_1, C) \in \mathcal{A}_1 \times \mathcal{C}_h$. Step (3) for the pair $(x, z) \in (C_1, C)$ ensures that either $\delta_H(x, z) \leq \delta_G(x, z) + 2$ or there is already a $C_1 \rightsquigarrow C$ path of length at most $|SP(x, z)|$. In either case there is a path $C_1 \rightsquigarrow z$ of length at most $\delta_G(x, z) + 2$. Since $\delta_H(s, C_1) \leq \delta_G(s, x)$ and $\delta_H(C_1, z) \leq \delta_G(x, z) + 2$, it follows that $\delta_H(s, v) \leq \delta_G(s, v) + 4$. ◀

Theorem 3.1 stated in Section 1 follows from Lemma 17 and Lemma 18 stated below.

▶ **Lemma 18.** *The size of the final subgraph H is $O(nh)$, where $h = \lceil (|S| n \log^2 n)^{2/9} \rceil$.*

5 ST-spanners: A trade-off result

In this section we present our algorithm to construct a sparse ST-spanner with additive stretch $2k$, for any integer $k \geq 1$. We first describe the algorithm and show that for any $(s, t) \in S \times T$, the final subgraph H has an s-t path of length at most $|SP(s, t)| + 2k$.

Initialization. We run the clustering step with an appropriate parameter h and the subgraph H is initialized to the post-clustering subgraph G_h. Then we augment H using $O(h)$ BFS trees so that for each shortest path $p \in \mathcal{F}$ with $\mathsf{cost}(p) \geq (n \log n)/h^2$, the subgraph H has a path of length at most $|p| + 2$ between p's endpoints.

Set the parameters $\ell = nh/(k\,|S|\,|T|)$ and $\alpha = (|S|\,|T|\,k \log n/h^3)^{1/k}$. Note that these parameters have been set so that $\alpha^k \ell = (n \log n)/h^2$. For $(s, t) \in S \times T$, if $\mathsf{cost}(SP(s, t)) \leq \ell$, then we buy $SP(s, t)$. So we now have to deal with approximating shortest paths $p \in \mathcal{F}$ with endpoints in $S \times T$ that satisfy $\ell < \mathsf{cost}(p) < \alpha^k \ell$.

Divide each such path p into *critical* subpaths as follows. Let $\alpha^{r-1}\ell < \mathsf{cost}(p) \leq \alpha^r \ell$ for some $r \in \{1, \ldots, k\}$. Then for each $j \in \{0, 1, \ldots, (r-1)\}$:

- let p_j = minimal suffix of p with $\lfloor \alpha^j \ell \rfloor$ edges missing in the post-clustering graph G_h.

Thus p can be split as $p' \,\|\, p_{r-1}$ for some prefix p'. Split p_{r-1} into $q_{r-1} \,\|\, q_{r-2} \,\|\, \cdots \,\|\, q_0$ as follows: let $q_0 = p_0$ and for $1 \leq j \leq r-1$, let q_j be the prefix of p_j obtained by removing p_{j-1} (a suffix of p_j) from p_j. For each j, let Q_j be the set of all q_j's. So each $p = SP(s, t)$ with $\mathsf{cost}(p) \geq \alpha^{r-1}\ell$ has a subpath $q_j \in Q_j$, for $j = 0, \ldots, r-1$.

Determine hitting sets $\mathcal{A}_0, \ldots, \mathcal{A}_{k-1}$ for these sets $Q_0, \ldots, Q_{k-1}$, respectively. In other words, $\mathcal{A}_j$ is a set of clusters such that for every $q_j \in Q_j$, there is at least one cluster in $\mathcal{A}_j$ that intersects q_j.

1. For each $s \in S$ and cluster $C_{j-1} \in \mathcal{A}_{j-1}$ (where $1 \leq j \leq k$) do
   - for every vertex x in C_{j-1}: if $\mathsf{cost}(SP(s, x)) \leq \alpha^j \ell$ and buying $SP(s, x)$ improves $\delta_H(s, C_{j-1})$, then buy $SP(s, x)$.

In order to see why this step is useful, consider $p = SP(s, t)$ such that $\alpha^{r-1}\ell < \mathsf{cost}(p) \leq \alpha^r \ell$ for some $1 \leq r \leq k$. As described above, we can write p as $p' \,\|\, q_{r-1} \,\|\, q_{r-2} \,\|\, \cdots \,\|\, q_0$. In each hitting set A_j (for $0 \leq j \leq r-1$), we would have at least one cluster that intersects q_j: let C_j be such a cluster. For each j, let u_j be the first vertex of cluster C_j on the path p, i.e., while traversing p from s to t, the first vertex of C_j that we encounter is u_j (see Figure 5).

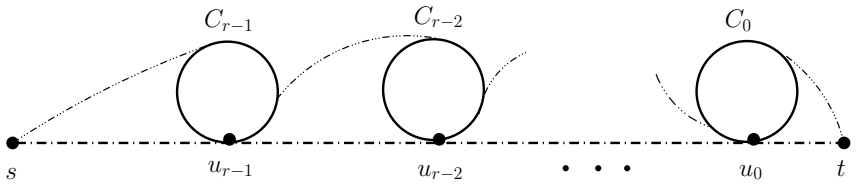


Figure 5 The subgraph H has a path of length at most $|SP(s, u_{r-1})|$ between s and C_{r-1}.

Consider the $s - u_{r-1}$ subpath of p. The shortest path $SP(s, u_{r-1})$, being a subpath of p, has cost at most $\alpha^r \ell$. Hence while processing the pair (s, C_{r-1}) when we consider $SP(s, u_{r-1})$ in Step (1) above, we either buy $SP(s, u_{r-1})$ or we already have $\delta_H(s, C_{r-1}) \leq \delta_G(s, u_{r-1})$. In other words, after Step (1) we have $\delta_H(s, C_{r-1}) \leq \delta_G(s, u_{r-1})$. We would similarly like to have $\delta_H(C_{i-1}, C_{r-2}) \leq |SP(u_{r-1}, u_{r-2})|$ and make $\delta_H(s, C_{r-2}) \leq \delta_G(s, u_{r-2}) + 2$ and so on. In order to accomplish this, we do as follows.

2. For each $s \in S$ and for $j = k - 1$ downto 1 do
- for every $(C_j, C_{j-1}) \in \mathcal{A}_j \times \mathcal{A}_{j-1}$ so that there is some $v \in C_{j-1}$ with $SP(s, v) \cap C_j \neq \emptyset$
  - for each $(x, y) \in C_j \times C_{j-1}$: if $\mathsf{cost}(SP(x, y)) \leq 2\alpha^j \ell$ and buying $SP(x, y)$ improves $\delta_H(s, C_{j-1})$, then buy $SP(x, y)$.

In the above step, corresponding to $j = r - 1$ and the vertex $s \in S$, we consider the pair (C_{r-1}, C_{r-2}) on the path p shown in Figure 5. Since there is a vertex $u_{r-2} \in C_{r-2}$ such that $SP(s, u_{r-2})$ intersects C_{r-1}, we run the innermost for loop of Step (2) for (C_{r-1}, C_{r-2}). Also $\mathsf{cost}(SP(u_{r-1}, u_{r-2})) \leq \mathsf{cost}(q_{r-1}) + \mathsf{cost}(q_{r-2}) \leq 2\alpha^{r-1}\ell$.

So either we already have $\delta_H(s, C_{r-2}) \leq \delta_G(s, u_{r-2}) + 2$ or we buy $SP(u_{r-1}, u_{r-2})$ and make $\delta_H(s, C_{r-2}) \leq \delta_H(s, C_{r-1}) + 2 + |SP(u_{r-1}, u_{r-2})|$, which is at most $\delta_G(s, u_{r-2}) + 2$ since $\delta_H(s, C_{r-1}) \leq \delta_G(s, u_{r-1})$ by the end of Step (1).

It is easy to see that for any $1 \leq i \leq r - 1$, if $\delta_H(s, C_i) \leq \delta_G(s, u_i) + 2(r - i - 1)$ when the index $j = i + 1$, then when the index $j = i$ and the pair (C_i, C_{i-1}) on path p gets considered, then $\delta_H(s, C_{i-1})$ becomes at most $\delta_H(s, C_i) + 2 + |SP(u_i, u_{i-1})|$ which is at most $\delta_G(s, u_{i-1}) + 2(r - i)$.

Thus at the end of Step (2), we have $\delta_H(s, C_0) \leq \delta_G(s, u_0) + 2(r - 1)$.

3. Finally for each $(s, t) \in S \times T$ and $C_0 \in \mathcal{A}_0$ such that C_0 intersects $SP(s, t)$ do
- for each $w \in C_0$: if $\mathsf{cost}(SP(w, t)) \leq \ell$ and buying $SP(w, t)$ improves $\delta_H(s, t)$, then buy $SP(w, t)$.

Thus in Step (3), when we consider the cluster C_0 and the pair (s, t), we either buy $SP(u_0, t)$ which ensures $\delta_H(s, t) \leq \delta_H(s, C_0) + 2 + |SP(u_0, t)| \leq \delta_G(s, t) + 2r$ or we already have $\delta_H(s, t) \leq \delta_G(s, t) + 2r$. Since $r \leq k$, it follows that H has an additive stretch of $2k$ for all distances in $S \times T$. This finishes the description of our algorithm and its correctness. Lemma 19 (proof omitted here) bounds the size of H.

▶ **Lemma 19.** *The final subgraph H has $O(nh)$ edges, where $h = \lceil k\sqrt{\log n} \cdot (|S|^{k+1}\,|T|)^{\frac{1}{2k+3}} \rceil$.*

We can now conclude Theorem 4 stated in Section 1, i.e., for any integer $k \geq 1$ and $S, T \subseteq V$, an ST-spanner with additive stretch $2k$ and size $\tilde{O}(n \cdot (|S|^\gamma |T|)^{1/(2\gamma+1)})$, where $\gamma = k + 1$, can be constructed in polynomial time.

By running the clustering step with $h = \lceil k\sqrt{\log n} \cdot (|S|^{k+1}\, n)^{1/(2k+4)} \rceil$ and taking $T = \{\text{cluster centers}\}$, Theorem 4 gives us an ST-spanner H of size $O(nh)$ and additive stretch $2k$. Since T is the set of cluster centers, it is easy to see that H is an $(S \times V)$-spanner with additive stretch $2k + 2$.

▶ **Corollary 20.** *For any integer $k \geq 1$ and subset $S \subseteq V$, an $(S \times V)$spanner with additive stretch $2t$ and size $\tilde{O}(n^{1+1/(2t+2)} \cdot |S|^{t/(2t+2)})$, where $t = k + 1$, can be constructed in polynomial time.*

Acknowledgments. Thanks to the reviewers for their helpful comments.

References

1. D. Aingworth, C. Chekuri, P. Indyk, and R. Motwani. Fast estimation of diameter and shortest paths (without matrix multiplication). *SIAM Journal on Computing*, 28(4):1167–1181, 1999.
2. I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete and Computational Geometry*, 9:81–100, 1993.
3. B. Awerbuch, B. Berger, L. Cowen, and D. Peleg. Near-linear time construction of sparse neighborhood covers. *SIAM Journal on Computing*, 28(1):263–277, 1998.
4. B. Awerbuch and D. Peleg. Routing with polynomial communication-space trade-off. *SIAM Journal on Computing*, 5(2):151–162, 1992.
5. S. Baswana and T. Kavitha. Faster algorithms for all-pairs approximate shortest paths in undirected graphs. *SIAM Journal on Computing*, 39(7):2865–2896, 2010.
6. S. Baswana, T. Kavitha, K. Mehlhorn, and S. Pettie. Additive spanners and (α, β)-spanners. *ACM Transactions on Algorithms*, 7(1), 2010.
7. S. Baswana and S. Sen. Approximate distance oracles for unweighted graphs in $o(n^2 \log n)$ time. *ACM Transactions on Algorithms*, 2(4):557–577, 2006.
8. S. Baswana and S. Sen. A simple linear time algorithm for computing a $(2k-1)$-spanner of $o(n^{1+1/k})$ size in weighted graphs. *Random Structures and Algorithms*, 30(4):532–563, 2007.
9. B. Bollobás, D. Coppersmith, and M. Elkin. Sparse distance preservers and additive spanners. *SIAM Journal on Discrete Math.*, 19(4):1029–1055, 2005.
10. S. Chechik. New additive spanners. In *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 498–512, 2013.
11. E. Cohen. Fast algorithms for constructing t-spanners and paths of stretch t. In *Proceedings of the 34th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 648–658, 1993.
12. D. Coppersmith and M. Elkin. Sparse source-wise and pair-wise distance preservers. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 660–669, 2005.
13. L. J. Cowen. Compact routing with minimum stretch. *Journal of Algorithms*, 28:170–183, 2001.
14. L. J. Cowen and C. G. Wagner. Compact roundtrip routing in directed networks. *Journal of Algorithms*, 50(1):79–95, 2004.
15. M. Cygan, F. Grandoni, and T. Kavitha. On pairwise spanners. In *Proceedings of the 30th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 209–220, 2013.
16. D. Dor, S. Halperin, and U. Zwick. All-pairs almost shortest paths. *SIAM Journal on Computing*, 29(5):1740–1759, 2004.
17. M. Elkin. Computing almost shortest paths. *ACM Transactions on Algorithms*, 1(2):283–323, 2005.
18. M. Elkin and D. Peleg. $(1+\epsilon, \beta)$-spanner construction for general graphs. *SIAM Journal on Computing*, 33(3):608–631, 2004.
19. C. Gavoille, D. Peleg, S. Perennes, and R. Raz. Distance labeling in graphs. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 210–219, 2001.

20 T. Kavitha and N. M. Varma. Small stretch pairwise spanners. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 601–612, 2013.

21 M. Parter. Bypassing Erdős' girth conjecture: Hybrid spanners and sourcewise spanners. In *Proceedings of the 41st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 608–619, 2014.

22 D. Peleg. Proximity-preserving labeling schemes. *Journal of Graph Theory*, 33(3):167–176, 2000.

23 D. Peleg and A. A. Schaffer. Graph spanners. *Journal of Graph Theory*, 13:99–116, 1989.

24 D. Peleg and J. D. Ullman. An optimal synchronizer for the hypercube. *SIAM Journal on Computing*, 18:740–747, 1989.

25 S. Pettie. Low distortion spanners. *ACM Transactions on Algorithms*, 6(1), 2009.

26 L. Roditty, M. Thorup, and U. Zwick. Deterministic constructions of approximate distance oracles and spanners. In *Proceedings of the 32nd Int. Colloq. on Automata, Languages, and Programming (ICALP)*, pages 261–272, 2005.

27 L. Roditty and U. Zwick. On dynamic shortest paths problems. In *Proceedings of the 12th Annual European Symposium on Algorithms (ESA)*, pages 580–591, 2004.

28 M. Thorup and U. Zwick. Approximate distance oracles. *Journal of the ACM*, 52(1):1–24, 2005.

29 D. P. Woodruff. Additive spanners in nearly quadratic time. In *Proceedings of the 37th Int. Colloq. on Automata, Languages, and Programming (ICALP)*, pages 463–474, 2010.