Yale University
Department of Computer Science

On Sparse Additive Spanners

Béla Bollobás [1 2 3 4]   Don Coppersmith [5]
Michael Elkin [6]

YALEU/DCS/TR-1305
September 2004

[1]Microsoft Research, One Microsoft Way, Redmond, WA 98122
[2]Department of Mathematical Sciences, University of Memphis, Memphis TN 38152
[3]Trinity College, Cambridge CB2 1TQ, UK
[4]Research supported by NSF grant DSM 9971788 and DARPA grant F33615-01-C-1900
[5]IBM Research, Yorktown Heights, NY, USA; dcopper@us.ibm.com
[6]Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot, 76100 Israel. elkin@wisdom.weizmann.ac.il.

On Sparse Additive Spanners

Béla Bollobás [*†‡§] Don Coppersmith [¶] Michael Elkin [‖]

Abstract

In this manuscript we prove the existence of arbitrarily sparse additive spanners with non-trivial additive error terms for general unweighted undirected graphs.

[*]Microsoft Research, One Microsoft Way, Redmond, WA 98122

[†]Department of Mathematical Sciences, University of Memphis, Memphis TN 38152

[‡]Trinity College, Cambridge CB2 1TQ, UK

[§]Research supported by NSF grant DSM 9971788 and DARPA grant F33615-01-C-1900

[¶]IBM Research, Yorktown Heights, NY, USA; dcopper@us.ibm.com

[‖]Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot, 76100 Israel. elkin@wisdom.weizmann.ac.il.

1 Introduction

A subgraph G' of an undirected graph $G = (V, E)$ is its *additive β-spanner* if for any pair of vertices $u, w \in V$, $d_{G'}(u, w) \leq d_G(u, w) + \beta$. (The same definition applies to Steiner graphs as well.) It is known [13, 15] that for any unweighted undirected n-vertex graph there exists an additive 2-spanner with $O(n^{3/2})$ edges, and an additive Steiner 4-spanner with $O(n^{4/3})$ edges. However, to the best of our knowledge, no construction of additive spanners with $o(n^{3/2})$ edges or Steiner additive spanners with $o(n^{4/3})$ edges are known.

It is implicit in [15] that the existence of a D-preserver with m edges for a graph implies the existence of an additive $O(D)$-spanner with the same number of edges for the same graph. Hence, our aforementioned results concerning D-preservers imply that for any n-vertex graph and for any $\delta > 0$ there exists an additive $O(n^{1-\delta})$-spanner with $O(n^{1+\delta})$ edges. We improve upon this and devise a construction of additive $O(2^{1/\delta} n^{(1-\delta)\frac{\lceil 1/\delta \rceil - 2}{\lceil 1/\delta \rceil - 1}})$-spanners with $O(n^{1+\delta})$ edges for any graph and any $\delta > 0$. In particular, this implies a construction of additive $O(n^{1/4+\epsilon/2})$-spanners with $O(n^{3/2-\epsilon})$ edges and $O(n^{1/3})$-spanners with $O(n^{4/3})$ edges. We remark that for additive Steiner spanners we achieve slightly better results.

2 The proof

In the following sequel we present a construction of additive $O(2^{1/\delta} n^{(1-\delta)\frac{\lceil 1/\delta \rceil - 2}{\lceil 1/\delta \rceil - 1}})$-spanners with $O(n^{1+\delta})$ edges for general unweighted undirected graphs.

First, let us introduce some notation. For any vertex set $C \subseteq V$ let $Diam(C) = \max\{d_G(u, w) \mid u, w \in C\}$. Also, for any vertex set C and a node $v \in C$ denote $Rad(C, v) = \max\{d_G(v, u) \mid u \in C\}$. Note that for any vertex set C and any $v \in C$, $Diam(C) \leq 2Rad(C, v)$.

We start with presenting procedure for constructing a partition of entire graph into disjoint small-diameter vertex sets, later on referred as *clusters*. The procedure accepts as input two parameters, the graph G and a number $0 < \delta < 1$. It outputs a partition $\mathcal{G}$ of the graph G into clusters of diameter not greater than $O(1/\delta)$, and a subset of edges $H \subseteq E$ containing the union of the breadth-first-search spanning trees of the clusters, along with at least one edge per each pair of clusters that are neighboring in G. We will later argue that $|H| = O(n^{1+\delta})$. The procedure is a generalization of a similar procedure in [21], in which $\delta = 1/\kappa$ for some integer $\kappa = 2, 3, \ldots$. The procedure starts with marking all the nodes as uncovered (i.e., putting all the nodes into the vertex set U). Then it picks an arbitrary uncovered node that has at least n^δ uncovered neighbors and constructs a cluster that contains this node and all its unclustered neighbors. Then it checks whether the newly constructed cluster has a lot of neighbors (specifically, whether the number of its neighbors is at least n^δ multiplied by the size of the cluster). If it has, all these neighbors are merged into the cluster and the same check is performed on the bigger cluster. Once this stopping condition becomes false, the procedure picks some arbitrary node that is still uncovered and constructs a cluster around him in an analogous manner. At the end we are left with only those uncovered nodes that do not have enough uncovered neighbors to create a cluster around them. For each such node the procedure constructs a singleton cluster that contains only this node. The formal description of Procedure *Ground_Part* follows.

Procedure *Ground_Part*

Input: G, $0 < \delta < 1$. **Output**: $\mathcal{G}$, $H \subseteq E$.

1. $H \leftarrow \emptyset \; ; \mathcal{G} \leftarrow \emptyset$;
2. $U \leftarrow V$; /* Set of uncovered nodes - initially the entire vertex set. */
3. **While** $\exists v \in U$ s.t. $deg_U(v) \geq n^\delta$ **do**
   (a) $C \leftarrow \{v\} \cup \{u \in U \mid (v,u) \in E\}$; $H \leftarrow \{(v,u) \in E \mid u \in U\}$;
   (b) **While** $|C \cup \{u \in U \setminus C \mid \exists z \in C \text{ s.t. } (z,u) \in E\}| \geq n^\delta |C|$ **do**
      i. $C' \leftarrow C \cup \{u \in U \setminus C \mid \exists z \in C \text{ s.t. } (z,u) \in E\}$;
      ii. **For** every $u \in U \setminus C$ s.t. $\exists z \in C$ s.t. $(z,u) \in E$ **do**
      Pick one arbitrary $z \in C$ with $(z,u) \in E$ and set $H \leftarrow H \cup \{(z,u)\}$;
      iii. $C \leftarrow C'$;
   (c) **For** every $u \in U \setminus C$ s.t. $\exists z \in C$ s.t. $(z,u) \in E$ **do**
   Pick one arbitrary $z \in C$ with $(z,u) \in E$ and set $H \leftarrow H \cup \{(z,u)\}$;
   (d) $U \leftarrow U \setminus C$; $\mathcal{G} \leftarrow \mathcal{G} \cup \{C\}$;
4. **For** every $v \in U$ **do** /* singleton clusters */
   (a) $C \leftarrow \{v\}$; $\mathcal{G} \leftarrow \mathcal{G} \cup \{C\}$;
   (b) $H \leftarrow H \cup \{(v,u) \mid u \in U \setminus C, (v,u) \in E\}$;
5. **return**$(\mathcal{G}, H)$;

Note that every cluster constructed at step 4 of Procedure *Ground_Part* contains precisely one node. We later on refer these clusters as *singleton clusters*. For any singleton cluster $C = \{v\}$ denote
$v = center(C)$ and $rad(C) = rad(C, center(C)) = rad(C, v) = 0$.

Note that every cluster constructed at step 3 contains $\Omega(n^\delta)$ nodes. Later on we refer to these clusters as *non-singleton* ones. Note that every iteration of the main loop (step 3 of Procedure *Ground_Part*) constructs precisely one non-singleton cluster. Let $center(C)$ denote the node v that was chosen at the iteration of the main loop on which the cluster C was constructed. All the rest of the clusters constructed by Procedure *Ground_Part* contain $\Omega(n^\delta)$ nodes. Later on we refer to these clusters as *non-singleton* ones. Intuitively, every non-singleton cluster C is constructed around some vertex v. This vertex is denoted $v = center(C)$. Let $rad(C) = rad(C, center(C)) = rad(C, v)$.

In particular, the above discussion implies that

Lemma 2.1 *For any cluster $C \in \mathcal{G}$, either $|C| = 1$ or $|C| = \Omega(n^\delta)$.*

Definition 2.2 *A collection of clusters $\mathcal{Z} = \{C_i\}$ such that $C_i \subseteq V$ for any $C_i \in \mathcal{Z}$, and such that any pair of clusters $C_i, C_j \in \mathcal{Z}$ such that $C_i \neq C_j$ satisfy $C_i \cap C_j = \emptyset$, is called a* partition of the vertex set $\bigcup_{C_i \in \mathcal{Z}} C_i$*. If the latter vertex set $\bigcup_{C_i \in \mathcal{Z}} C_i$ is a strict subset of V then $\mathcal{Z}$ is called a* partial partition*; otherwise it is called a* partition.

In the next lemma we show that the clusters constructed by Procedure *Ground_Part* have bounded radii.

Lemma 2.3 *For any cluster $C \in \mathcal{G}$, $rad(C) \leq \lceil 1/\delta \rceil - 1$. Also, $|C| = \Omega(n^{rad(C)\cdot\delta})$.*

Proof: For singleton clusters the claim is obvious.

For a non-singleton cluster $C \in \mathcal{G}$, its radius is equal to the number of iterations of the internal *while* loop (step 3b of Procedure *Ground_Part*) on the iteration of the main loop on which the cluster C was constructed, plus 1. (The extra 1 is from step 3a.)

Since at the beginning of step 3b, $|C| \geq n^\delta + 1$, and each iteration of the internal *while* loop increases the size of C by at least a factor of n^δ, it follows that after j iterations,
$|C| \geq (n^\delta + 1)n^{\delta j} = n^{\delta(j+1)} + n^{\delta j}$. Since $n \geq |C|$, it follows that $n^{\delta(j+1)} < n$. Thus, $j+1 < 1/\delta$. I.e.,
$j < 1/\delta - 1 \leq \lceil 1/\delta \rceil - 1$. Hence, $j \leq \lceil 1/\delta \rceil - 2$. Thus, $rad(C) = j + 1 \leq \lceil 1/\delta \rceil - 1$,
and $|C| = \Omega(n^{\delta(j+1)}) = \Omega(n^{\delta\cdot rad(C)})$. ∎

Next, we analyze the cardinality of the edge set H constructed by Procedure *Ground_Part*.

Lemma 2.4 *Let $(\mathcal{G}, H)$ be the output of the invocation of Procedure Ground_Part with input (G, δ), for a graph $G = (V, E)$, and $0 < \delta < 1$. Let $\tilde{G} = (\tilde{V}, \tilde{E})$ be the supergraph induced by the partition $\mathcal{G}$ of graph G. I.e., $\tilde{V} = \mathcal{G}$, $\tilde{E} = \{(C_1, C_2)$ s.t. $d_G(C_1, C_2) = 1\}$. Then $|\tilde{E}|, |H| = O(n^{1+\delta})$.*

Proof: For every cluster $C \in \mathcal{G}$, let us associate with it all the superedges $(C, C_0) \in \tilde{E}$ and all the edges $(u, u_0) \in (C \times C_0) \cap H$, such that C_0 was constructed after C. Denote by $\tilde{A}(C)$ (resp., $A(C)$) the set of superedges (resp., edges of H) associated with C. Note that every superedge is associated with precisely one cluster. I.e., $\tilde{E} = \bigcup_{C \in \mathcal{G}} \tilde{A}(C)$. Therefore, $|\tilde{E}| = \sum_{C \in \mathcal{G}} |\tilde{A}(C)|$. Similarly, every edge of H is associated with precisely one cluster. Thus, $|H| = \sum_{C \in \mathcal{G}} |A(C)|$. Note also that when the construction of C was completed, the total number r of nodes in U that are neighbors of one of the nodes of C was $O(|C| n^{\delta})$. Further, any cluster C_0 that is a neighbor of C in $\tilde{G}$ and was constructed after C must contain at least one of these nodes. Therefore, $|\tilde{A}(C)| = O(|C| n^{\delta})$. Note also that the number of edges that were inserted into H on step 3c or on step 4b of the iteration on which the cluster C was constructed is precisely r. Thus, $|A(C)| = O(|C| n^{\delta})$. It follows that both $|\tilde{E}|$ and $|H|$ are at most

$$\sum_{C \in \mathcal{G}} O(|C| n^{\delta}) \ = \ O(n^{\delta}) \sum_{C \in \mathcal{G}} |C| \ = \ O(n^{1+\delta}). \qquad \blacksquare$$

Finally, let us argue that

Lemma 2.5 *The output collection $\mathcal{G}$ of Procedure Ground_Part is a partition.*

Proof: Consider some pair of clusters $C_1, C_2 \in \mathcal{G}$. Suppose, without loss of generality, that C_1 was constructed before C_2. Denote by U_0 the set of nodes that were in U at the beginning of the iteration of the main loop of Procedure *Ground_Part* on which C_2 was constructed. Note that $U_0 \cap C_1 = \emptyset$. Note also that $C_2 \subseteq U_0$. It follows that $C_1 \cap C_2 = \emptyset$ for any pair of different clusters $C_1, C_2 \in \mathcal{G}$.

Note that for any vertex $u \in V$, it was either inserted to some cluster $C \in \mathcal{G}$ at step 3 of Procedure *Ground_Part*, or the cluster $C = \{u\}$ was inserted into $\mathcal{G}$ at step 4. Thus $\mathcal{G}$ is a partition. $\blacksquare$

We later on refer to the partition $\mathcal{G}$ constructed by Procedure *Ground_Part* as *ground partition.*

We will also make use of Procedure *Threshold_Spanner*, that is proven in [7]. This procedure accepts as input an unweighted undirected graph G and a number $0 < \nu < 1$ and returns a subgraph $H \subseteq E$ such that for any $u, w \in V$ such that $d_G(u, w) \geq n^{\nu}$ it holds that $d_H(u, w) = d_G(u, w)$. Furthermore, $|H| = O(n^{2-\nu})$.

We next describe Algorithm *Additive_Spanner* that constructs spanners with arbitrarily small size and purely additive error; this error, however, depends on n as n^{ϵ_0} for some $0 < \epsilon_0 < 1$. The algorithm is actually a variant of the algorithm for constructing $(1+\epsilon, \beta(\epsilon,\kappa))$-spanners of size $O(n^{1+1/\kappa})$ due to [15]. Like the algorithm of [15], Algorithm *Additive_Spanner* iteratively gets rid of small clusters by unifying them into bigger ones (later on called *superclusters*) and interconnecting the pairs of close small clusters that cannot be unified into a supercluster. The clusters are considered close if the distance between them is at most a certain threshold. The value of this threshold determines the diameters of the superclusters that are constructed. The value of the threshold must always be significantly greater than the maximal diameter of the small clusters treated on the specific iteration. Since the constructed superclusters act as small clusters on the next iteration of the algorithm, the value of the threshold increases correspondingly. Since the value of the threshold is a lower bound on the additive error of the spanner constructed by the algorithm, the algorithm of [15] uses constant (but growing) values of the threshold on all the iterations. However, it turns out that setting the threshold to be n^{ϵ_0} for some $0 < \epsilon_0 < 1$ and using the result from [7] that says that all the distances above $n^{1-\delta}$ can be preserved by a subgraph of size $O(n^{1+\delta})$) it is possible to get a spanner with arbitrarily small size and purely additive error.

The formal description of Algorithm *Additive_Spanner* follows.
Denote

$$\tilde{\delta} = \frac{1-\delta}{\lceil 1/\delta \rceil - 1} \ . \qquad (1)$$

For any two clusters $C_1, C_2 \subseteq V$ denote by P_{C_1,C_2} one of the shortest paths between them in G. For any path P let $E(P)$ denote the edge set of the path P. Procedure *SuperCluster* is described later on.

Algorithm *Additive_Spanner*

Input: $G = (V, E)$, $0 < \delta < 1$. **Output:** H

1. $(\mathcal{G}, H) \leftarrow Ground\_Part(G, \delta)$;
2. $\mathcal{R}_0 \leftarrow \{C \in \mathcal{G} \mid rad(C) = 0\}$; $\tilde{V}_0 \leftarrow \mathcal{R}_0$; $\tilde{E}_0 \leftarrow \{(C_1, C_2) \in \mathcal{R}_0^2 \mid d_G(C_1, C_2) = 1\}$;
3. $H_0 \leftarrow Threshold\_Spanner(G, 1-\delta)$; $H \leftarrow H \cup H_0$;
4. (a) Construct supergraph $\tilde{G}_1 = (\tilde{V}_1, \tilde{E}_1)$ with
   $\tilde{V}_1 = \{C \in \mathcal{G} \mid rad(C) = 1\}$,
   $\tilde{E}_1 = \{(C_1, C_2) \in \tilde{V}_1^2 \mid d_G(C_1, C_2) \leq n^{\tilde{\delta}}\}$;

(b) **For** $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$ **do**

i. $(\mathcal{R}_i, \mathcal{Z}_{i+1}, H) \leftarrow SuperCluster(\tilde{G}_i, n^{\delta}, H)$;

ii. $H \leftarrow H \cup \bigcup_{(C_1,C_2) \in \mathcal{R}_i^2 \cap \tilde{E}_i} E(P_{C_1,C_2})$;

iii. Construct $\tilde{G}_{i+1} = (\tilde{V}_{i+1}, \tilde{E}_{i+1})$ with
$\tilde{V}_{i+1} = \mathcal{Z}_{i+1} \cup \{C \in \mathcal{G} \mid rad(C) = i+1\}$,
$\tilde{E}_{i+1} = \{(C_1, C_2) \in \tilde{V}_{i+1}^2 \mid d_G(C_1, C_2) \leq n^{(i+1)\tilde{\delta}}\}$;

5. (a) $\mathcal{R}_{\lceil 1/\delta \rceil - 1} \leftarrow \tilde{V}_{\lceil 1/\delta \rceil - 1}$;

(b) $H \leftarrow H \cup \bigcup_{(C_1,C_2) \in \mathcal{R}^2_{\lceil 1/\delta \rceil - 1} \cap \tilde{E}_{\lceil 1/\delta \rceil - 1}} E(P_{C_1,C_2})$;

6. **return**(H);

We next describe Procedure *SuperCluster*. The procedure accepts as input a supergraph $\tilde{G} = (\tilde{V}, \tilde{E})$ over a graph $G = (V, E)$, a degree parameter deg and an edge set $H \subseteq E$. The graph $\tilde{G}$ is a supergraph in the sense that every vertex of $\tilde{G}$ corresponds to some vertex set in the graph G. The procedure outputs a set $\mathcal{R} \subseteq \tilde{V}$, referred later on as the $\mathcal{R}$-*output collection* of the procedure, a partial partition $\mathcal{Z}'$ of $\tilde{V}$, referred later on as the $\mathcal{Z}$-*output collection* of the procedure, and an edge set H' such that $H \subseteq H' \subseteq E$. Similarly to the way Procedure *Ground_Part* operates, Procedure *SuperCluster* starts with marking all the supernodes as uncovered (inserting them into $\mathcal{R}$) and then picks an arbitrary uncovered supernode that has a sufficient number of neighbors (the threshold is determined by the the degree parameter deg), and constructs a supercluster out of this cluster and all its uncovered neighbors. However, unlike Procedure *Ground_Part*, Procedure *SuperCluster* does not try to insert into this supercluster its neighbors even if it has a big neighborhood.

The formal description of Procedure *SuperCluster* follows. For any subset $\mathcal{R} \subseteq \tilde{V}$, denote $\tilde{G}(\mathcal{R})$ the subgraph of $\tilde{G}$ induced by $\mathcal{R}$.

Procedure *SuperCluster*
Input: $\tilde{G} = (\tilde{V}, \tilde{E})$, deg, $H \subseteq E$. **Output**: $\mathcal{R} \subseteq \tilde{V}$, partial partition $\mathcal{Z}'$ of $\tilde{V}$, H'.

1. $\mathcal{Z}' \leftarrow \emptyset$; $\mathcal{R} \leftarrow \tilde{V}$; $H' \leftarrow H$;

2. **While** $\exists C \in \mathcal{R}$ s.t. $deg_{\tilde{G}(\mathcal{R})}(C) \geq deg$ **do**

(a) $H' \leftarrow H' \cup \bigcup_{(C,C_0) \in \tilde{E}} E(P_{C,C_0})$;

(b) $C' \leftarrow C \cup \bigcup_{(C,C_0) \in \tilde{E}} C_0$;

(c) $\mathcal{Z}' \leftarrow \mathcal{Z}' \cup \{C'\}$;

(d) $\mathcal{R} \leftarrow \mathcal{R} \setminus \{C\} \setminus \{C_0 \mid (C, C_0) \in \tilde{E}\}$;

3. **return**$(\mathcal{R}, \mathcal{Z}', H')$;

Next, we are going to analyze the properties of Procedure *SuperCluster*. First, we need some definitions.

Definition 2.6 *For an unweighted undirected graph $G = (V, E)$, its partial partition $\mathcal{Z}$ and a fixed positive integer D, the graph $\tilde{G} = (\tilde{V}, \tilde{E})$, with $\tilde{V} = \mathcal{Z}$, $\tilde{E} = \{(C_1, C_2) \mid C_1, C_2 \in \tilde{V}; d_G(C_1, C_2) \le D\}$ is called the* supergraph over G induced by $\mathcal{Z}$ and D.

Also, for a partial partition $\mathcal{Z}$ of a graph G, let $minsize(\mathcal{Z}) = \min\{|C| \mid C \in \mathcal{Z}\}$, and $maxDiam(\mathcal{Z}) = \max\{Diam(C)| \mid C \in \mathcal{Z}\}$. The following lemma summarizes the main properties of Procedure *SuperCluster*.

Lemma 2.7 *Let $\tilde{G} = (\tilde{V}, \tilde{E})$ be a supergraph over a graph $G = (V, E)$ induced by some partial partition $\mathcal{Z}$ and distance threshold $D \ge 1$. Consider an invocation of Procedure SuperCluster with input $(\tilde{G}, deg, H)$ for some fixed degree parameter deg and edgeset $H \subseteq E$. Let $(\mathcal{R}, \mathcal{Z}', H')$ be its output. Then*
(1) $minsize(\mathcal{Z}') \ge minsize(\mathcal{Z}) \cdot deg$; (2) $maxDiam(\mathcal{Z}') \le 2D + 3maxDiam(\mathcal{Z})$; (3) Every cluster $C' \in \mathcal{Z}'$ is a union of some non-empty set of clusters of $\mathcal{Z}$; (4) $\mathcal{R} \subseteq \mathcal{Z}$; (5) For every cluster $C \in \mathcal{Z}$, either $C \in \mathcal{R}$ and for any cluster $C' \in \mathcal{Z}'$, $C \nsubseteq C'$, or $C \notin \mathcal{R}$, but there exists a cluster $C' \in \mathcal{Z}'$ such that $C \subseteq C'$. Also, for any $C_1 \in \mathcal{Z}'$, $C_2 \in \mathcal{R}$, $C_1 \cap C_2 = \emptyset$. (6) $\mathcal{Z}'$ is a partial partition of G ;
(7) $|H'| \le |H| + O(D|\tilde{V}|)$.

Proof: (1): Follows since by the stopping condition of the *while* loop of Procedure *SuperCluster*, every cluster $\mathcal{Z}'$ is a union of at least *deg + 1* clusters of $\mathcal{Z}$.

(2): Consider some $C' \in \mathcal{Z}'$. It is formed at step 2b of Procedure *SuperCluster*. Let $C \in \mathcal{Z}$ be the cluster that was picked on this iteration of the *while* loop. Consider some $u, w \in C'$. Let C_u be the cluster of $\mathcal{Z}$ that contains u, and C_w be the one that contains w. Note that $d_G(C, C_u) \le D$, because either $C_u = C$ and then $d_G(C, C_u) = 0$ or $(C, C_u) \in \tilde{E}$ implying $d_G(C, C_u) \le D$, since $\tilde{G}$ is a supergraph over G induced by $\mathcal{Z}$ and D. Analogously, $d_G(C, C_w) \le D$. Let $v_0 \in C$, $u_0 \in C_u$ be the nodes such that $d_G(v_0, u_0) = d_G(C, C_u)$. Analogously, let $v_0' \in C$, $w_0 \in C_w$ be the nodes such that $d_G(v_0', w_0) = d_G(C, C_w)$. Then,

$$\begin{aligned} d_G(u, w) &\le d_G(u, u_0) + d_G(u_0, v_0) + d_G(v_0, v_0') + \\ &+ d_G(v_0', w_0) + d_G(w_0, w) \\ &\le maxDiam(\mathcal{Z}) + D + maxDiam(\mathcal{Z}) + D + maxDiam(\mathcal{Z}) \\ &\le 2D + 3maxDiam(\mathcal{Z}) \ , \end{aligned}$$

completing the proof of (2).

(3): Follows by step 2b of Procedure *SuperCluster*.

(4): Follows because $\tilde{V} = \mathcal{Z}$ by Definition 2.6, and by steps 1 and 2d of Procedure *SuperCluster*.

(5): Follows because every cluster C of $\mathcal{Z}$ was initially inserted into $\mathcal{R}$, and by steps 2b and 2d of Procedure *SuperCluster*, the cluster either remained in $\mathcal{R}$ or was merged into some cluster of $\mathcal{Z}'$. This also proves that

$$\bigcup_{C \in \mathcal{Z}} C = \bigcup_{C \in \mathcal{R}} C \cup \bigcup_{C' \in \mathcal{Z}'} C' \ .$$

The same argument proves also the second assertion of (5). (6): Note that no cluster $C \in \mathcal{Z}$ is inserted into two different clusters of $\mathcal{Z}'$, since only clusters of $\mathcal{R}$ may be inserted into a cluster of $\mathcal{Z}'$, and once they are inserted into a cluster of $\mathcal{Z}'$ they are removed from $\mathcal{R}$ (steps 2b, 2c, 2d).

Now the claim follows from the assumption that $\mathcal{Z}$ is a partial partition.

(7): Consider the collection $\mathcal{F} \subseteq \tilde{E}$ of superedges $(C, C_0) \in \tilde{E}$ such that the edgesets $E(P_{C,C_0})$ were inserted into H' on steps 2a of Procedure *SuperCluster*. Note that all the superedges whose corresponding paths were inserted into H' on some specific iteration of the *while* loop of Procedure *SuperCluster* form a star.

Also, for any star $\{(C, C_i)\}_{i \in I}$ of superedges such that the edgeset $\bigcup_{i \in I} E(P_{C,C_0})$ was inserted into H' on some specific iteration of the *while* loop, at the end of this iteration the set of clusters $\{C\} \cup \bigcup_{i \in I} \{C_i\}$ was removed from $\mathcal{R}$. Also, for any path P_{C,C_0} such that $E(P_{C,C_0})$ is inserted into H', at the step of Procedure *SuperCluster* on which $E(P_{C,C_0})$ is inserted into H', both clusters C and C_0 still belong to $\mathcal{R}$.

It follows that $\mathcal{F}$ is a forest in $\tilde{G}$. I.e., $|\mathcal{F}| \leq |\tilde{V}|$. Each superedge of $\mathcal{F}$ corresponds to a path of length at most D that was inserted into H'. Thus, the overall number of edges inserted into H' is $O(|\tilde{V}|D)$. ∎

Next, using the properties of Procedures *SuperCluster* and *Ground_Part* summarized in previous lemmas, we analyze Algorithm *Additive_Spanner*. The following lemma establishes a lower bound on the sizes of the clusters constructed at different iterations of Algorithm *Additive_Spanner*.

Lemma 2.8 *For* $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 1$, $minsize(\tilde{V}_i) = \Omega(n^{i\delta})$.

Proof: The proof is by induction on i.

For the induction base, note that $\tilde{V}_1 = \{C \in \mathcal{G} \mid rad(C) = 1\}$. By Lemma 2.3, $minsize(\tilde{V}_1) \geq n^{\delta}$.

For the induction step, we assume the assertion for $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$. and prove it for $i+1$.

By step 4(b)iii of Algorithm *Additive_Spanner*, $\tilde{V}_{i+1} = \mathcal{Z}_{i+1} \cup \{C \in \mathcal{G} \mid rad(C) = i+1\}$. Thus, $minsize(\tilde{V}_{i+1}) = \min\{minsize(\mathcal{Z}_{i+1}), minsize(\{C \in \mathcal{G} \mid rad(C) = i+1\})\}$. By Lemma 2.3,
$minsize(\{C \in \mathcal{G} \mid rad(C) = i+1\}) = \Omega(n^{(i+1)\delta})$.

By Lemma 2.7(1), and since all the invocations of Procedure *SuperCluster* are with degree parameter $deg = n^{\delta}$ (see step 4(b)i of Algorithm *Additive_Spanner*), it follows that $minsize(\mathcal{Z}_{i+1}) \geq minsize(\tilde{V}_i) \cdot n^{\delta}$. By induction hypothesis, $minsize(\tilde{V}_i) = \Omega(n^{i\delta})$. Hence, $minsize(\mathcal{Z}_{i+1}) = \Omega(n^{i\delta}) \cdot \Omega(n^{\delta}) = \Omega(n^{(i+1)\delta})$. Hence,
$minsize(\tilde{V}_{i+1}) = \Omega(n^{(i+1)\delta})$. ▮

In the following lemma we analyze the diameters of the clusters constructed at different iterations of Algorithm *Additive_Spanner*.

Lemma 2.9 *For any $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 1$, $maxDiam(\tilde{V}_i) = O(n^{(i-1)\tilde{\delta}})$.*

Proof: The proof is by induction on i.

For $i = 1$, $\tilde{V}_1 = \{C \in \mathcal{G} \mid rad(C) = 1\}$. Thus, $maxDiam(\tilde{V}_1) = O(1)$, and we are done.

For the induction step we assume the assertion for $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$, and prove it for $i+1$.

Note that $maxDiam(\tilde{V}_{i+1}) = \max\{maxDiam(\mathcal{Z}_{i+1}), maxDiam(\{C \in \mathcal{G} \mid rad(C) = i+1\})\}$.
Also, $maxDiam(\{C \in \mathcal{G} \mid rad(C) = i+1\}) = O(1/\delta)$. Also, by Lemma 2.7(2), $maxDiam(\mathcal{Z}_{i+1}) \leq 2D + 3maxDiam(\tilde{V}_i)$, where D is the distance threshold using which the supergraph $\tilde{G}_i$ was formed. Recall that on ith iteration $\tilde{V}_i = \mathcal{Z}_i \cup \{C \in \mathcal{G} \mid rad(C) = i\}$, and the distance threshold is $D = n^{i\tilde{\delta}}$. Hence,

$$\begin{aligned} maxDiam(\mathcal{Z}_{i+1}) &= 2D + 3\max\{maxDiam(\mathcal{Z}_i), maxDiam(\{C \in \mathcal{G} \mid rad(C) = i\})\} \\ &\leq 2n^{i\tilde{\delta}} + 3\max\{n^{(i-1)\tilde{\delta}}, O(1/\delta)\} = O(n^{i\tilde{\delta}}) . \end{aligned}$$ ▮

Corollary 2.10 *For any $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 1$, $maxDiam(\mathcal{R}_i) \leq maxDiam(\tilde{V}_i) = O(n^{(i-1)\tilde{\delta}})$.*

Proof: Follows directly from Lemmas 2.9 and 2.7(4). ▮

Corollary 2.11 *For any $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 1$, $minsize(\mathcal{R}_i) \geq minsize(\tilde{V}_i) = \Omega(n^{i\delta})$.*

Proof: Follows directly from Lemmas 2.8 and 2.7(4). ▮

In the next lemma we argue that clusters of $\mathcal{R}_i$ have small number of $\mathcal{R}_i$-neighbors in the graph $\tilde{G}_i$.

Lemma 2.12 *For any $i = 0, 1, \ldots, \lceil 1/\delta \rceil - 1$, and for any cluster $C \in \mathcal{R}_i$, $deg_{\tilde{G}_i(\mathcal{R}_i)}(C) = O(n^\delta)$, where $\tilde{G}_i(\mathcal{R}_i)$ is the subgraph of $\tilde{G}_i$ induced by $\mathcal{R}_i$.*

Proof: For $i = 0$, $\mathcal{R}_0 = \{C \in \mathcal{G} \mid |C| = 1\}$. Suppose for contradiction that some singleton cluster $\{v\} = C \in \mathcal{R}_0$ has more than n^δ neighboring clusters of $\mathcal{R}_0$, i.e., more than n^δ neighboring singleton clusters. However, all the singleton clusters were constructed at step 4 of Procedure *Ground_Part*. Therefore, at step 3 of the procedure all the nodes that are contained in these singleton clusters were in U. It follows that after the last iteration of the main *while* loop (step 3) of Procedure *Ground_Part* the node v was in U, and furthermore, it had more than n^δ neighbors in U. But then the stopping condition of the *while* loop was satisfied, contradicting the assumption that the procedure left the loop. Therefore, for any $C \in \mathcal{R}_0$, $deg_{\tilde{G}_0(\mathcal{R}_0)}(C) = O(n^\delta)$.

For $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$, recall that $\mathcal{R}_i$ is the $\mathcal{R}$-output collection of Procedure *SuperCluster* invoked on the ith iteration of Algorithm *Additive_Spanner*. Similar argument to the one that we applied regarding the main loop of Procedure *Ground_Part* in the case $i = 0$ can now be applied to the main *while* loop of Procedure *SuperCluster* (step 2). It follows that for any $C \in \mathcal{R}_i$, for $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$, $deg_{\tilde{G}_i(\mathcal{R}_i)}(C) = O(n^\delta)$.

Finally, by Corollary 2.11, for $i = \lceil 1/\delta \rceil - 1$, $minsize(\mathcal{R}_{\lceil 1/\delta \rceil - 1}) = \Omega(n^{(\lceil 1/\delta \rceil - 1)\delta}) = \Omega(n^{1-\delta})$.
Hence, $|\mathcal{R}_{\lceil 1/\delta \rceil - 1}| = O(n^\delta)$. Therefore, for any $C \in \mathcal{R}_{\lceil 1/\delta \rceil - 1}$, $deg_{\tilde{G}_{\lceil 1/\delta \rceil - 1}(\mathcal{R}_{\lceil 1/\delta \rceil - 1})}(C) = O(n^\delta)$. ▮

The following lemma establishes the upper bound on the size of the spanner that Algorithm *Additive_Spanner* outputs.

Lemma 2.13 $|H| = O(\delta^{-1} n^{1+\delta})$.

Proof: The subgraph H_0 which is the output of Procedure *Threshold_Spanner* is of size $O(n^{1+\delta})$. Recall that H_0 satisfies that for any pair of vertices $u, w \in V$ such that $d_G(u, w) \geq n^{1-\delta}$ holds
$d_{H_0}(u, w) = d_G(u, w)$.

By Lemma 2.4, Procedure *Ground_Part* also inserts into H at most $O(n^{1+\delta})$ edges.

On iterations $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$, Procedure *SuperCluster* is invoked, and it inserts into the spanner H some edges at steps 2a. By Lemma 2.8, $minsize(\tilde{V}_i) = \Omega(n^{i\delta})$. Therefore,

$|\tilde{V}_i| = O(n^{1-i\delta})$. By step 4(b)iii, the distance threshold with which the supergraph $\tilde{G}_i$ is formed is $O(n^{i\tilde{\delta}})$. Note that $\tilde{\delta} = \frac{1-\delta}{\lceil 1/\delta \rceil - 1} \leq \delta$. Hence, by Lemma 2.7(7), the number of edges inserted by Procedure *SuperCluster* on iteration i is $O(n^{1-i\delta} n^{i\tilde{\delta}}) = O(n)$. Since the total number of iterations is at most $O(1/\delta)$, Procedure *SuperCluster* inserts on iterations $i = 1, 2, \ldots, \lceil 1/\delta \rceil - 2$ at most $O(n\delta^{-1})$ edges into the spanner.

In addition, on each iteration of the main loop of Algorithm *Additive_Spanner*, edges are inserted into the spanner H at steps 4(b)ii. However, note that by Lemma 2.12, $deg_{\tilde{G}_i(\mathcal{R}_i)}(C) = O(n^{\delta})$. Thus, by Corollary 2.11, on iteration i, the number of paths inserted on this step into H is at most

$$\begin{aligned} O(|\mathcal{R}_i| \max_{C \in \mathcal{R}_i} deg_{\mathcal{R}_i}(C)) &= O(\frac{n}{minsize(\mathcal{R}_i)} \cdot n^{\delta}) \\ &= O(n^{1-(i-1)\delta}) \ . \end{aligned}$$

Every path that is inserted at step 4(b)ii of Algorithm *Additive_Spanner* on iteration i corresponds to an edge of $\tilde{E}_i$, and thus its length is not greater than $O(n^{i\tilde{\delta}}) = O(n^{i\delta})$.

Hence, altogether the number of edges inserted into H at step 4(b)ii of iteration i is at most $O(n^{1-(i-1)\delta} n^{i\delta}) = O(n^{1+\delta})$. Since the number of iterations is $O(1/\delta)$, it follows that the number of edges inserted on steps 4(b)ii of the main loop of Algorithm *Additive_Spanner* is $O(\delta^{-1} n^{1+1/\delta})$.

Finally, at step 5b of Algorithm *Additive_Spanner* all the paths between clusters of $\mathcal{R}_{\lceil 1/\delta \rceil - 1} = \tilde{V}_{\lceil 1/\delta \rceil - 1}$ are inserted. By Corollary 2.11,
$minsize(\mathcal{R}_{\lceil 1/\delta \rceil - 1}) = \Omega(n^{(\lceil 1/\delta \rceil - 1)\delta})$. I.e., $|\mathcal{R}_{\lceil 1/\delta \rceil - 1}| = O(n^{1+\delta-\lceil 1/\delta \rceil \delta}) = O(n^{\delta})$.

Thus, the number of paths between the clusters of $\mathcal{R}_{\lceil 1/\delta \rceil - 1}$ is $O(n^{2+2\delta-2\lceil 1/\delta \rceil \delta})$. Each such path contains $O(n^{1-\delta}) = O(n^{(\lceil 1/\delta \rceil - 1)\tilde{\delta}})$ edges. Hence, the number of edges inserted on step 5b of Algorithm *Additive_Spanner* is

$$\begin{aligned} O(|\mathcal{R}_{\lceil 1/\delta \rceil - 1}|^2 n^{(\lceil 1/\delta \rceil - 1)\tilde{\delta}} &= O(n^{2+2\delta-2\lceil 1/\delta \rceil \delta + 1 - \delta}) \\ &= O(n^{3+\delta-2\delta\lceil 1/\delta \rceil}) = O(n^{1+\delta}) \ . \end{aligned}$$

Hence, $|H| = O(n^{1+\delta}/\delta)$. ▮

The next lemma establishes a close relation between the partitions $\mathcal{Z}_{i+1}$, $i = 1, 2, 3, \ldots, \lceil 1/\delta \rceil - 2$, returned by Procedure *SuperCluster* on ith iteration of the main loop of Algorithm *Additive_Spanner* (step 4(b)i), and the ground partition $\mathcal{G}$.

Lemma 2.14 *For any $i = 1, 2, 3, \ldots, \lceil 1/\delta \rceil - 2$, $\mathcal{Z}_{i+1}$ is a collection of clusters C that satisfy $C = \bigcup_{C_0 \in Z} C_0$, where $\emptyset \neq Z \subseteq \{C_0 \in \mathcal{G} \mid rad(C_0) \leq i\}$.*

Proof: The proof is by induction on i.

The base of the induction is $i = 1$. $\mathcal{Z}_2$ is the $\mathcal{Z}$-output collection of the first invocation of Procedure *SuperCluster*. Its input partition is $\tilde{V}_1 = \{C \in \mathcal{G} \mid rad(C) = 1\}$.

Hence, by Lemma 2.7(3), every cluster $C' \in \mathcal{Z}_2$ is a union of a non-empty set of clusters of $\tilde{V}_1$.

For the induction step recall that $\mathcal{Z}_{i+1}$ is the $\mathcal{Z}$-output collection of the ith invocation of Procedure *SuperCluster*. Its input partition is $\tilde{V}_i = \mathcal{Z}_i \cup \{C \in \mathcal{G} \mid rad(C) = i\}$. By induction hypothesis, any $C \in \mathcal{Z}_i$ is a union of a non-empty set of clusters C_0 with radius $rad(C_0) \leq i-1$. Thus, for any $C \in \tilde{V}_i$, C is a union of a non-empty set of clusters C_0 of radius $rad(C_0) \leq i$. Thus, by Lemma 2.7(3), every cluster of $\mathcal{Z}_{i+1}$ is a union of a non-empty set of clusters C_0 of radius $rad(C_0) \leq i$. ▮

In the following lemma we analyze the structure of the partitions $\mathcal{R}_i$ that are constructed throughout Algorithm *Additive_Spanner*.

Lemma 2.15 *For any* $i = 0, 1, \ldots, \lceil 1/\delta \rceil - 2$, $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \tilde{V}_{i+1}$ *is a partition of the vertex set* $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq i+1\}$.

Proof: The proof is by induction on i.

For the induction base we argue that $\mathcal{R}_0 \cup \tilde{V}_1$ is a partition of the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq 1\}$. Indeed, by steps 2 and 4a of Algorithm *Additive_Spanner*, $\mathcal{R}_0 = \{C \in \mathcal{G} \mid rad(C) = 0\}$, $\tilde{V}_1 = \{C \in \mathcal{G} \mid rad(C) = 1\}$. Now the claim follows since $\mathcal{G}$ is a partition (by Lemma 2.5).

For the induction step, assume that for some $i = 0, 1, \ldots, \lceil 1/\delta \rceil - 3$, $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \tilde{V}_{i+1}$ is a partition of the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq i+1\}$. We next argue that

1. The clusters of $\mathcal{R}_{i+1}$ are disjoint.
2. The clusters of $\tilde{V}_{i+2}$ are disjoint.
3. For any $C \in \mathcal{R}_{i+1}$ and $C' \in \tilde{V}_{i+2}$, $C \cap C' = \emptyset$.
4. For any $C \in \mathcal{R}_{i+1}$, $C' \in R_j$, $j = 0, 1, \ldots, i$, $C \cap C' = \emptyset$.
5. For any $C' \in \tilde{V}_{i+2}$, $C \in \mathcal{R}_j$, $j = 0, 1, \ldots, i$, $C \cap C' = \emptyset$.
6. For any cluster $C \in \mathcal{G}$ of radius $rad(C) \leq i+2$, there exists a cluster $C' \in \bigcup_{j=0}^{i+1} \mathcal{Z}_j \cup \mathcal{R}_{i+2}$ such that $C \subseteq C'$.

This would complete the proof of the lemma, since (1)- (6) imply that $\bigcup_{j=0}^{i+1} \mathcal{Z}_j \cup \mathcal{R}_{i+2}$ is a partition of the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq i+2\}$.

To argue (1), note that by Lemma 2.7(4). $\mathcal{R}_{i+1} \subseteq \tilde{V}_{i+1}$. Note also that $\tilde{V}_{i+1}$ is a partial partition by the induction hypothesis. Therefore, $\mathcal{R}_{i+1}$ is a partial partition as well, proving the claim.

To argue (2), we first note that $\tilde{V}_{i+2} = \mathcal{Z}_{i+2} \cup \{C \in \mathcal{G} \mid rad(C) = i+2\}$. By Lemma 2.14, for any $C \in \mathcal{Z}_{i+2}$, C is a non-empty union of clusters $C_0 \in \mathcal{G}$ of radius $rad(C_0) \leq i+1$. Note that $\mathcal{Z}_{i+2}$ is the $\mathcal{Z}$-output partition of the $(i+1)$st invocation of Procedure *SuperCluster*, whose input is $\tilde{V}_{i+1}$. $\tilde{V}_{i+1}$ is a partial partition by the induction hypothesis. Therefore, $\mathcal{Z}_{i+2}$ is a partial partition by Lemma 2.7(6).

Since $\mathcal{G}$ is a partition (by Lemma 2.5), clusters of the collection $\{C \in \mathcal{G} \mid rad(C) = i+2\}$ are disjoint.

Finally, for any cluster $C' \in \mathcal{Z}_{i+2}$, C' is a union of a non-empty set of clusters $C_0 \in \mathcal{G}$ of $rad(C_0) \leq i+1$, by Lemma 2.14. Thus, for any cluster $C' \in \mathcal{Z}_{i+2}$, and $C \in \mathcal{G}$ such that $rad(C) = i+2$, $C \cap C' = \emptyset$. This completes the proof of (2).

To argue (3), we first note that $\tilde{V}_{i+2} = \mathcal{Z}_{i+2} \cup \{C \in \mathcal{G} \mid rad(C) = i+2\}$.

Observe that $\mathcal{Z}_{i+2}$ is the $\mathcal{Z}$-output collection of the $(i+1)$st invocation of Procedure *SuperCluster*, and $\mathcal{R}_{i+1}$ is the $\mathcal{R}$-output collection of the same invocation. Therefore, the disjointness of their clusters follows by Lemma 2.7(5), and since the input collection $\tilde{V}_i$ is a partition by the induction hypothesis.

We still have to argue that for any $C \in \mathcal{G}$ with $rad(C) = i+2$, and for any $C' \in \mathcal{Z}_{i+2} \cup \mathcal{R}_{i+1}$, $C \cap C' = \emptyset$. Note that $\mathcal{R}_{i+1} \subseteq \tilde{V}_{i+1} = \mathcal{Z}_{i+1} \cup \{C \in \mathcal{G} \mid rad(C) = i+1\}$, and $\mathcal{Z}_{i+1}$ consists of clusters that are non-empty unions of clusters of $\mathcal{G}$ of radius at most i, by Lemma 2.14. Thus, by Lemma 2.14 and since by Lemma 2.5 $\mathcal{G}$ is a partition, both $\mathcal{Z}_{i+2}$ and $\mathcal{R}_{i+1}$ consist of clusters that are non-empty unions of clusters of radius at most $i+1$, and we are done.

To argue (4), note that $\mathcal{R}_{i+1}$ is the $\mathcal{R}$-output partition of the $(i+1)$st invocation of Procedure *SuperCluster*, and its input partition is $\tilde{V}_{i+1}$. Thus $\mathcal{R}_{i+1} \subseteq \tilde{V}_{i+1}$. Furthermore, by induction hypothesis, $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \tilde{V}_{i+1}$ is a partial partition, and thus $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \mathcal{R}_{i+1}$ is a partial partition as well.

To argue (5), we note that by Lemma 2.7(3) every cluster of $\mathcal{Z}_{i+2}$ is a union of a non-empty set of clusters $C_0 \in \tilde{V}_{i+1}$. By induction hypothesis, $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \tilde{V}_{i+1}$ is a partial partition, and so $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \mathcal{Z}_{i+2}$ is a partial partition as well.

We still have to argue that for any cluster $C \in \mathcal{G}$ of radius $i+2$ and for any cluster $C' \in \mathcal{R}_j$, for $j = 0, 1, \ldots, i$, $C \cap C' = \emptyset$.

This is because $\mathcal{R}_j \subseteq \mathcal{Z}_j \cup \{C \in \mathcal{G} \mid rad(C) = j\}$, and by Lemma 2.14, $\mathcal{Z}_j$ is a collection of clusters such that each of them is a non-empty union of clusters of $\mathcal{G}$ of radius at most $j-1$. Thus, $\mathcal{R}_j$ is a collection of clusters such that each of them is a non-empty union of clusters of radius smaller than i. On the other hand, $rad(C) = i+2$, and $\mathcal{G}$ is a partition, and so we are done.

To argue (6), recall that by induction hypothesis, $\bigcup_{j=0}^{i} \mathcal{R}_j \cup \tilde{V}_{i+1}$ is a partition of the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq i+1\}$.

Consider $\bigcup_{j=0}^{i+1} \mathcal{R}_j \cup \tilde{V}_{i+2}$. By the induction hypothesis, $\tilde{V}_{i+1}$ is a partial partition. Recall that it is the input partition of the $(i+1)$st invocation of Procedure *SuperCluster*, and $\mathcal{R}_{i+1}$ and $\mathcal{Z}_{i+2}$ are its output partitions.

By Lemma 2.7(5), for any $C \in \tilde{V}_{i+1}$ either $C \in \mathcal{R}_{i+1}$ or there exists a cluster $C' \in \mathcal{Z}_{i+2}$ such that $C \subseteq C'$. Note that by (1)-(5), $\bigcup_{j=0}^{i+1} \mathcal{R}_j \cup \tilde{V}_{i+2}$ is a partition. It follows that the vertex set
$\{v \in C \mid C \in \mathcal{G}, rad(C) \leq i+1\}$ is covered by the partition $\bigcup_{j=0}^{i+1} \mathcal{R}_j \cup \mathcal{Z}_{i+2}$. Also, $\tilde{V}_{i+2} = \mathcal{Z}_{i+2} \cup \{C \in \mathcal{G} \mid rad(C) = i+2\}$. However, the set of clusters $\{C \in \mathcal{G} \mid rad(C) = i+2\}$ covers the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) = i+2\}$. Therefore, the partition $\bigcup_{j=0}^{i+1} \mathcal{R}_j \cup \mathcal{Z}_{i+2} \cup \{C \in \mathcal{G} \mid rad(C) = i+2\} = \bigcup_{j=0}^{i+1} \mathcal{R}_j \cup \tilde{V}_{i+2}$ covers the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq i+2\}$. ∎

Corollary 2.16 $\bigcup_{j=0}^{\lceil 1/\delta \rceil - 1} \mathcal{R}_j$ *is a partition of* G.

Proof: By Lemma 2.15, $\bigcup_{j=0}^{\lceil 1/\delta \rceil - 2} \mathcal{R}_j \cup \tilde{V}_{\lceil 1/\delta \rceil - 1}$ is a partition of the vertex set $\{v \in C \mid C \in \mathcal{G}, rad(C) \leq \lceil 1/\delta \rceil - 1\}$.

By Lemma 2.3, the latter vertex set is equal to V. Now the assertion of the corollary follows since $\tilde{V}_{\lceil 1/\delta \rceil - 1} = \mathcal{R}_{\lceil 1/\delta \rceil - 1}$, by step 5a of Algorithm *Additive_Spanner*. ∎

Next we analyze the relation between the distances in the spanner H and the distances in the original graph G.

Consider some pair of vertices $u, w \in V$. If $d_G(u,w) \geq n^{1-\delta}$ then $d_H(u,w) = d_G(u,w)$, since $H_0 \subseteq H$. So, from now on we assume that $d_G(u,w) < n^{1-\delta}$.

Lemma 2.17 *Let* $u, w \in V$ *be a pair of vertices such that* $d_G(u,w) \leq n^{i\tilde{\delta}}$ *such that one of the shortest paths between them in* G, $P_{u,w}$, *contains only vertices clustered in clusters of* $\left(\bigcup_{j=0}^{i} \mathcal{R}_j\right)$,
$i = 0, 1, \ldots, \lceil 1/\delta \rceil$. *Then,* $d_H(u,w) \leq d_G(u,w) + (2^i + 2^{i-1} - 2) c_0 n^{(i-1)\tilde{\delta}}$,

where $maxDiam(\mathcal{R}_i) \le c_0 n^{(i-1)\tilde{\delta}}$ *(see Corollary 2.10;* c_0 *is a small universal constant).*

Proof: The proof is by induction on i. For $i = 0$, by the assumption of the lemma, $P_{u,w}$ is $\mathcal{R}_0$-clustered. I.e., for any node $z \in P_{u,w}$, the node is clustered in some cluster C such that $rad(C) = 0$. Recall that
$\mathcal{R}_0 = \{C \in \mathcal{G} \mid rad(C) = 0\}$, i.e., $\mathcal{R}_0$ is the set of singleton clusters of $\mathcal{G}$. Finally, for any pair of neighboring clusters C_1, C_2 of $\mathcal{G}$, at least one edge $(u, w) \in (C_1 \times C_2) \cap E$ was inserted into H (step 3c of Procedure *Ground_Part*). Therefore, $d_H(u, w) = d_G(u, w)$.

For the induction step, assume the assertion for $i = 0, 1, \ldots, \lceil 1/\delta \rceil - 2$.

Consider some pair of vertices $u, w \in V$ such that $d_G(u, w) \le n^{(i+1)\tilde{\delta}}$, and some shortest path $P_{u,w}$ between them that contains only vertices clustered in $\bigcup_{j=0}^{i+1} \mathcal{R}_j$.

If $P_{u,w}$ contains no node of $\mathcal{R}_{i+1}$, then the claim follows by the induction hypothesis. Specifically, we partition $P_{u,w}$ into segments of length $n^{i\tilde{\delta}}$ (except the last one that may be shorter). On each segment, by induction hypothesis, the distance in the spanner H is at most by $(2^i + 2^{i-1} - 2)c_0 n^{(i-1)\tilde{\delta}}$ greater than the distance in G. Since there are at most $n^{\tilde{\delta}}$ segments, it follows that
$d_H(u, w) \le d_G(u, w) + (2^i + 2^{i-1} - 2)c_0 n^{i\tilde{\delta}}$.

If only one vertex $z \in P_{u,w}$ is clustered in some cluster C of $\mathcal{R}_{i+1}$, then in addition to the error of the previous case, the shortest path between u and w in H may be longer by
$Diam(C) \le maxDiam(\mathcal{R}_{i+1}) \le c_0 n^{i\tilde{\delta}}$. I.e., in this case
$d_H(u, w) \le d_G(u, w) + c_0(2^i + 2^{i-1} - 1)n^{i\tilde{\delta}}$.

Finally, consider the case when at least two vertices along $P_{u,w}$ are $\mathcal{R}_{i+1}$-clustered. It is helpful to visualize the path as if it goes from left to right, from u to w. Let u' be the leftmost $\mathcal{R}_{i+1}$-clustered vertex of $P_{u,w}$, and w' be the rightmost of them. Let $C_{u'}$ be the cluster containing u', and $C_{w'}$ be the cluster containing w'. Note that $C_{u'}, C_{w'} \in \mathcal{R}_{i+1}$. Since $d_G(C_{u'}, C_{w'}) \le n^{(i+1)\tilde{\delta}}$, the shortest path between them was inserted into the spanner H either on step 4(b)ii (if $i < \lceil 1/\delta \rceil - 2$) or on step 5b (if $i = \lceil 1/\delta \rceil - 2$) of Algorithm *Additive_Spanner*.

So,

$$d_H(u', w') \le d_G(u', w') + 2maxDiam(\mathcal{R}_{i+1}) \le d_G(u'w') + 2c_0 n^{i\tilde{\delta}} \ . \tag{2}$$

Denote the segment from u to u' of $P_{u,w}$ by P_1, and the segment from w' to w by P_2. Note that both paths P_1 and P_2 are $\left(\bigcup_{j=0}^{i} \mathcal{R}_j\right)$-clustered. Partition P_1 into segments of length $n^{i\tilde{\delta}}$ (except the last segment that might be shorter). Consider one such segment from u_1'' to u_2''. By the induction hypothesis, $d_H(u_1'', u_2'') \le d_G(u_1'', u_2'') + (2^i + 2^{i-1} - 2)c_0 n^{(i-1)\tilde{\delta}}$. Since

$d_G(u,u') \le n^{(i+1)\tilde{\delta}}$, there are at most $n^{\tilde{\delta}}$ such segments between u and u'. Thus,

$$d_H(u,u') \le d_G(u,u') + (2^i + 2^{i-1} - 2)c_0 n^{i\tilde{\delta}} \ . \tag{3}$$

Analogously,

$$d_H(w',w) \le d_G(w',w) + (2^i + 2^{i-1} - 2)c_0 n^{i\tilde{\delta}} \ . \tag{4}$$

Hence, by (2), (3) and (4), it follows that

$$\begin{aligned} d_H(u,w) &\le d_G(u,w) + 2\cdot(2^i + 2^{i-1} - 2)c_0 n^{i\tilde{\delta}} + 2c_0 n^{i\tilde{\delta}} \\ &= d_G(u,w) + (2^{i+1} + 2^i - 2)c_0 n^{i\tilde{\delta}} \ . \quad \blacksquare \end{aligned}$$

Corollary 2.18 *For any pair of nodes $u,w \in V$, $d_H(u,w) \le d_G(u,w) + O(2^{1/\delta} n^{(1-\delta)\frac{\lceil 1/\delta\rceil - 2}{\lceil 1/\delta\rceil - 1}})$.*

Proof: Recall that if $d_G(u,w) \ge n^{1-\delta}$ than $d_H(u,w) = d_G(u,w)$, since $H_0 \subseteq H$, and H_0 is the output of Procedure *Threshold_Spanner* that was invoked on G and $(1-\delta)$ (see step 3 of Algorithm *Additive_Spanner*), and using the distance-size preservation theorem from [7].

Hence, in the rest of the proof we consider the case $d_G(u,w) \le n^{1-\delta}$. Note also that for any pair of vertices $u,w \in V$, the shortest path $P_{u,w}$ between them is $\left(\bigcup_{j=0}^{\lceil 1/\delta\rceil - 1} \mathcal{R}_j\right)$-clustered, since by Corollary 2.16, $\left(\bigcup_{j=0}^{\lceil 1/\delta\rceil - 1} \mathcal{R}_j\right)$ is a partition of G. Furthermore, by the definition of $\tilde{\delta}$ (see (1)),
$d_G(u,w) \le n^{1-\delta} = n^{(\lceil 1/\delta\rceil - 1)\tilde{\delta}}$. So, the pair of vertices u,w satisfy the condition of Lemma 2.17 with $i = \lceil 1/\delta\rceil - 1$. Thus,

$$\begin{aligned} d_H(u,w) &\le d_G(u,w) + (2^{\lceil 1/\delta\rceil - 1} + 2^{\lceil 1/\delta\rceil - 2} - 2)c_0 n^{(\lceil 1/\delta\rceil - 2)\tilde{\delta}} \\ &= d_G(u,w) + O(2^{1/\delta} n^{(1-\delta)\frac{\lceil 1/\delta\rceil - 2}{\lceil 1/\delta\rceil - 1}}) \ . \quad \blacksquare \end{aligned}$$

To summarize,

Theorem 2.19 *For any unweighted undirected n-vertex graph $G = (V,E)$ and for any fixed $0 < \delta < 1$ there exists a subgraph, constructible in polynomial time, $H \subseteq E$ of size $|H| = O(n^{1+\delta}/\delta)$ such that for any pair of vertices $u,w \in V$,
$d_H(u,w) \le d_G(u,w) + O(2^{1/\delta} n^{(1-\delta)\frac{\lceil 1/\delta\rceil - 2}{\lceil 1/\delta\rceil - 1}})$.*

Note that for $\delta = 1/2$, $\lceil 1/\delta\rceil - 2 = 0$, and so $d_H(u,w) \le d_G(u,w) + O(1)$, implying an alternative to [13, 15] construction of additive $O(1)$-spanner with $O(n^{3/2})$ edges. However, in our construction the constant error is 4 and not 2 as in [13, 15].

For $1/3 < \delta = 1/2 - \epsilon < 1/2$, $(\epsilon < 1/6)$, $\frac{\lceil 1/\delta\rceil - 2}{\lceil 1/\delta\rceil - 1} = 1/2$, and
$d_H(u,w) \le d_G(u,w) + O(n^{(1/2)(1-\delta)}) = d_G(u,w) + O(n^{1/4+\epsilon/2})$. So, in particular, this is a

construction of spanners with $o(n^{3/2})$ edges with *purely additive* error (which is, however, polynomially-dependent on n ($O(n^{1/4+\epsilon/2})$ when using $O(n^{3/2-\epsilon})$ edges)). Note that for $\delta = 1/3$, this is a construction of additive $O(n^{1/3})$-spanners with $O(n^{4/3})$ edges. For $1/4 \leq \delta = 1/3 - \epsilon < 1/3$ (i.e., $0 < \epsilon < 1/12$),
$d_H(u,w) \leq d_G(u,w) + O(n^{(2/3)(1-\delta)}) = d_G(u,w) + O(n^{4/9+(2/3)\epsilon})$.

Observe that the exponent of the additive error is not a continuous function of δ (that determines the number of edges used by the subgraph, $O(n^{1+\delta})$). Specifically, there are discontinuity points in $\delta = 1/\kappa$ for $\kappa = 2, 3, \ldots$. Indeed, for $\delta = 1/\kappa$, $d_H(u,w) \leq d_G(u,w) + O(n^{(\kappa-2)/\kappa})$. However, for $\delta = 1/\kappa - \epsilon > 1/(\kappa+1)$ ($0 < \epsilon < 1/\kappa - 1/(\kappa+1) = \frac{1}{\kappa(\kappa+1)}$), $\lceil 1/\delta \rceil = \kappa + 1$, and

$$\begin{aligned} d_H(u,w) &\leq d_G(u,w) + O(n^{(1-\delta)(\kappa-1)/\kappa}) = d_G(u,w) + O(n^{((\kappa-1)/\kappa+\epsilon)(\kappa-1)/\kappa}) \\ &= d_G(u,w) + O(n^{(\kappa-1)^2/\kappa^2+\epsilon(\kappa-1)/\kappa}) \ . \end{aligned}$$

It follows that when $\delta \searrow 1/(\kappa+1)$ the additive error of the spanner tends to $O(n^{\frac{\kappa-1}{\kappa+1}})$. Indeed,

$$\lim_{\epsilon \to \frac{1}{\kappa(\kappa+1)}} \left(\frac{(\kappa-1)^2}{\kappa^2} + \epsilon(\kappa-1)/\kappa \right) = \frac{\kappa-1}{\kappa+1} \ . \tag{5}$$

I.e., the exponent of the additive error is a right-continuous function of δ (but, not left-continuous in $\delta = 1/\kappa$, $\kappa = 2, 3, \ldots$).

References

[1] I. Althöfer, G. Das, D. Dobkin, D. Joseph and J. Soares, On sparse spanners of weighted graphs, *Discrete & Computational Geometry* **9**, (1993), 81–100.

[2] B. Awerbuch and D. Peleg, Sparse partitions, *Proc. 31st IEEE Symp. on Foundations of Computer Science*, 503–513, October 1990.

[3] B. Awerbuch and D. Peleg. Network synchronization with polylogarithmic overhead, In *Proc. 31st Symp. on Foundations of Computer Science*, pp. 514-522, October, 1990.

[4] B. Awerbuch, B. Berger, L. Cowen, D. Peleg, Near-linear time construction of sparse neighborhood covers, *SIAM Journal of Computing*, Vol. 28, **1**, 263-277, 1998.

[5] Bollobás, B., *Modern Graph Theory*, Graduate Texts in Mathematics, vol. **184**, Springer-Verlag, New York, 1998, xiv+394 pp. (OR, Second Edition, 2002)

[6] Bollobás, B., *Random Graphs*, Second Edition, Cambridge Studies in Advanced Mathematics **73**, Cambridge University Press, Cambridge, 2001, xviii+498 pp.

[7] B. Bollobás, D. Coppersmith and M. Elkin. Sparse Distance Preservers and Additive Spanners. In *Proc. 14th Annual ACM-SIAM Symp. on Discr. Algorithms*, pp. 412-423, USA, MD, Baltimore, 2003.

[8] B. Chandra, G. Das, G. Narasimhan and J. Soares, New sparseness results on graph spanners, *Proc. 8th ACM Symp. on Computat. Geometry*, 192–201, 1992.

[9] L. P. Chew, There is a planar graph almost as good as the complete graph, *2nd Symp. on Computational Geometry*, pages 169–177, 1986.

[10] E. Cohen, Fast Algorithms for constructing t-spanners and paths of stretch t, in *Proc. 34th IEEE Symp. on Foundations of Computer Science*, IEEE, Piscataway, NJ, 1993, 648-658.

[11] E. Cohen, Polylog-time and near-linear work approximation scheme for undirected shortest paths, in *Proc. of 26th ACM Symp. on Theory of Computation*, pp. 16-26, 1994.

[12] D. P. Dobkin, S. J. Friedman and K.J. Supowit, Delaunay graphs are almost as good as complete graphs, *Proc. 31st IEEE Symp. on Foundations of Computer Science*, 1987, 20–26.

[13] D. Dor, S. Halperin, U. Zwick, All pairs almost shortest paths, *SIAM Journal on Computing*, 29 (2000), pp. 1740-1759.

[14] M. L. Elkin, Computing Almost Shortest Paths, in *Proc. 20th ACM Symp. on Principles of Distributed Computing*, pp.53-63, Newport, Rhode Island, August, 2001.

[15] M. L. Elkin and D. Peleg. $(1+\epsilon,\beta)$-Spanner Constructions for General Graphs, *Proc. 33rd ACM Symp. on Theory of Computing*, pp. 173-182, Crete, Greece, July, 2001.

[16] T. Feder and R. Motwani. Clique Partitions, Graph Compression and Speeding-up Algorithms. Special Issue for the STOC conference, Journal of Computer and System Sciences, 51 (1995):261-272.

[17] C. Gavoille, D. Peleg, S. Perennes, R. Raz. Distance labeling in graphs. *Proc. to 12th Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 210-219, Washington, DC, USA, January, 2001.

[18] R. L. Graham, B. L. Rothschield, J. H. Spencer, *Ramsey Theory*, Second Edition, Wiley-Interscience Series in Discrete Mathematics and Optimization, A Wiley-Interscience Publication, John Wiley & Sons, Inc., New York, 1990, xii+196 pp.

[19] D. Peleg, *Distributed computing: a locality-sensitive approach*, SIAM, Philadelphia, PA, 2000.

[20] D. Peleg, Proximity-preserving labeling schemes, *J. Graph Theory*, 33:167-176, 2000.

[21] D. Peleg and A. Schäffer. Graph Spanners, *Journal of Graph Theory* **13** (1989), 99-116.

[22] D. Peleg and J. D. Ullman, An optimal synchronizer for the hypercube, *SIAM J. Computing* **18**, (1989), 740–747.

[23] M. Thorup, Oct. 2001, private communication.

[24] M. Thorup, Compact Oracles for Reachability and Approximate Distances on Planar Digraphs, *Proc. 42nd Annual Symp. on Foundations of Computer Science*, Las Vegas, Nevada, Oct., 2001.

[25] M. Thorup and U. Zwick, Approximate Distance Oracles, *Proc. 33rd ACM Symp. on Theory of Computing*, pp. 183-192, Crete, Greece, July, 2001.

[26] J. D. Ullman, M. Yannakakis, High-Probability Parallel Transitive-Closure-Algorithms. SIAM J. Comput. 20(1): 100-125 (1991).