

The Emergence of Sparse Spanners and Greedy Well-Separated Pair Decomposition

Jie Gao and Dengpan Zhou

Department of Computer Science, Stony Brook University,
Stony Brook, NY 11794, USA {jgao, dpzhou}@cs.sunysb.edu

Abstract. A spanner graph on a set of points in \mathbb{R}^d provides shortest paths between any pair of points with lengths at most a constant factor of their Euclidean distance. A spanner with a sparse set of edges is thus a good candidate for network backbones, as in transportation networks and peer-to-peer network overlays. In this paper we investigate new models and aim to interpret why good spanners ‘emerge’ in reality, when they are clearly built in pieces by agents with their own interests and the construction is not coordinated. Our main result is to show that the following algorithm generates a $(1 + \varepsilon)$ -spanner with a linear number of edges. In our algorithm, the points build edges at an *arbitrary* order. A point p will only build an edge pq if there is no existing edge $p'q'$ with p' and q' at distances no more than $\frac{1}{4(1+\varepsilon)} \cdot |pq|$ from p, q respectively. Eventually when all points finish checking edges to all other points, the resulted collection of edges forms a sparse spanner as desired. As a side product, the spanner construction implies a greedy algorithm for constructing linear-size well-separated pair decompositions that may be of interest on its own.

Key words: Spanner, Well-separated pair decomposition, Greedy algorithm

1 Introduction

A geometric graph G defined on a set of points $P \subseteq \mathbb{R}^d$ with all edges as straight line segments of weight equal to the length is called a *Euclidean spanner*, if for any two points $p, q \in P$ the shortest path in G has length at most $s \cdot |pq|$ where $|pq|$ is the Euclidean distance. The factor s is called the *stretch factor* of G and the graph G is called an s -spanner. Spanners with a sparse set of edges provide good approximations for the pairwise Euclidean distances and are good candidates for network backbones. Thus, there has been a lot of work on the construction of sparse Euclidean spanners in both the centralized [19, 33] and distributed settings [34].

In this paper we are interested in the emergence of good Euclidean spanners formed by uncoordinated agents. Many real-world networks, such as the transportation network and the Internet backbone network, are good spanners — one can typically drive from any city to any other city in the U.S. with the total travel distance at most a small constant times their straight line distance. The same thing happens with the Internet backbone graph as well. However, these large networks are not owned or built by any single authority. They are often assembled with pieces built by different governments or different ISPs, at different points in time. Nevertheless altogether they provide a

convenient sparse spanner. The work in this paper is motivated by this observation of the lack of coordination in reality and we would like to interpret why a good Euclidean spanner is able to ‘emerge’ from these agents incrementally.

Prior work that attempt to remove centralized coordination has been done, as in the network creation game [21, 15, 26, 4, 31], first introduced by Fabrikant *et al.* [21] to understand the evolution of network topologies maintained by selfish agents. A cost function is assigned to each agent, capturing the cost paid to build connections to others minus the benefit received due to the resulting network topology. The agents play a game by minimizing their individual costs and one is interested in the existence and the price of anarchy of Nash equilibria. Though being theoretically intriguing, there are two major open questions along this direction. First, the choice of cost functions is heuristic. Almost all past literatures use a unit cost for each edge and they deviate in how the benefit of ‘being connected to others’ is modeled. There is little understanding on what cost function best captures the reality yet small variation in the cost function may result in big changes in the network topologies at Nash equilibria. There is also not much understanding of the topologies at Nash equilibria, some of them are simplistic topologies such as trees or complete graphs, that do not show up often in the real world. It remains open whether there is a natural cost model with which the Nash equilibrium is a sparse spanner.

The game theoretic model also has limitations capturing the reality: selfish agents may face deadlines and have to decide on building an edge or not immediately; once an edge is built, it probably does not make sense to remove it (as in the case of road networks); an agent may not have the strategies of all other agents making the evaluation of the cost function difficult. In this paper, we take a different approach and ask whether there is any simple rule, with which each agent can determine on its own, and collectively build and maintain a sparse spanner topology without any necessity of coordination or negotiation. The simple rule serves as a ‘certificate’ of the sparse spanner property that warrants easy spanner maintenance under edge dynamics and node insertion. We believe such models and good algorithms under these models worth further exploration and this paper makes a first step along this line.

Our contribution. We consider in this paper the following model that abstracts the scenarios explained earlier. There are n points in the plane. Each point represents a separate agent and may consider to build edges from itself to other points. These decisions can happen at different points in time. When an agent p plans on an edge pq , p will only build it if whether there does not exist a ‘nearby’ edge $p'q'$ in the network, where $|pp'|$ and $|qq'|$ are within $\frac{1}{4(1+\varepsilon)} \cdot |p'q'|$ from p and q respectively. This strategy is very intuitive — if there is already a cross-country highway from Washington D.C. to San Francisco, it does not make economical sense to build a highway from New York to Los Angeles. We assume that each agent will eventually check on each possible edge from itself to all other points, but the order on who checks which edge can be *completely arbitrary*. With this strategy, the agents only make decisions with limited information and no agent has full control over how and what graph will be constructed. It is not obvious that this strategy will lead to a sparse spanner. It is not clear that the graph is even connected.

The main result in this paper is to prove that with the above strategy executed in *any* arbitrary order, the graph built at the end of the process is a sparse spanner:

- The stretch factor of the spanner is $1 + \varepsilon$.
- The number of edges is $O(n)$.
- The total edge length of the spanner is $O(|\text{MST}| \cdot \log \alpha)$, where α is the *aspect ratio*, i.e., the ratio of the distance between the furthest pair and the closest pair, and $|\text{MST}|$ is the total edge length of the minimum spanning tree of the point set.
- The degree of each point is $O(\log \alpha)$ in the worst case and $O(1)$ on average.

To explain how this result is proved, we first obtain as a side product the following *greedy* algorithm for computing a well-separated pair decomposition. A pair of two sets of points, (A, B) , is called *s-well-separated* if the smallest distance between any two points in A, B is at least s times greater than the diameters of A and B . An *s-well-separated pair decomposition* (*s-WSPD* for short) for P is a collection of *s-well-separated* pairs $\mathcal{W} = \{(A_i, B_i)\}$ such that for any pair of points $p, q \in P$ there is a pair $(A, B) \in \mathcal{W}$ with $p \in A$ and $q \in B$. The size of an *s-WSPD* is the number of point set pairs in \mathcal{W} . Well-separated pair decomposition (WSPD) was first introduced by Callahan and Kosaraju [12] and they developed algorithms for computing an *s-WSPD* with linear size for points in \mathbb{R}^d . Since then WSPD has found many applications in computing *k*-nearest neighbors, *n*-body potential fields, geometric spanners and approximate minimum spanning trees [9, 10, 12, 11, 6, 5, 32, 29, 24, 20].

So far there are two algorithms for computing optimal size WSPD, one in the original paper [12] and one in a later paper [22]. Both of them use a hierarchical organization of the points (e.g., the fair split tree in [12] and the discrete center hierarchy in [22]) and output the well-separated pairs in a recursive way. In this paper we show the following simple *greedy* algorithm also outputs an *s-WSPD* with linear size. We take an *arbitrary* pair of points p, q that is not yet covered in any existing well-separated pair, and consider the pair of subsets $(B_r(p), B_r(q))$ with $r = |pq|/(2s + 2)$ and $B_r(p)$ ($B_r(q)$) as the set of points of P within distance r from p (q). Clearly $(B_r(p), B_r(q))$ is an *s-well-separated* pair and all the pairs of points (p', q') with $p' \in B_r(p)$ and $q' \in B_r(q)$ are covered. The algorithm continues until all pairs of points are covered. We show that, no matter in which order the pairs are selected, the greedy algorithm will always output a linear number of well-separated pairs. Similarly, this greedy algorithm can be executed in an environment when coordination is not present, while the previous algorithms (in [12, 22]) cannot.

WSPD is deeply connected to geometric spanners. Any WSPD will generate a spanner if one puts an edge between an arbitrary pair of points p, q from each well-separated pair $(A, B) \in \mathcal{W}$ [6, 5, 32, 29]. The number of edges in the spanner equals the size of \mathcal{W} . In the other direction, the deformable spanner proposed in [22] implies a WSPD of linear size. The connection is further witnessed in this paper: our spanner emergence algorithm implies a WSPD generated in a greedy manner. Thus the well-separated pairs and spanner edges are one-to-one correspondence.

Last, this paper focuses on the Euclidean case when the points are distributed in the plane. The basic idea extends naturally to points in higher dimensions as well as metrics with constant doubling dimensions [25] (thus making the results applicable in

non-Euclidean settings), as the main technique involves essentially various forms of geometric packing arguments.

Applications. The results can be applied in maintaining nice network overlay topologies for P2P file sharing applications [30]. Such P2P overlay networks are often constructed in a distributed manner without centralized control, to achieve robustness, reliability and scalability. One important issue is reducing routing delay by making the overlay topology aware of the underlying network topology [14, 35, 28, 39, 40]. But all these work are heuristics without any guarantee. A spanner graph would be a good solution for the overlay construction, yet there is no centralized authority in the P2P network that supervises the spanner construction and the peers may join or leave the network frequently. The work in this paper initiates the study of the emergence of good spanners in the setting when there is little coordination between the peers and the users only need a modest amount of incomplete information of the current overlay topology.

We show that the spanner can be constructed under a proper model such that only $O(n \log \alpha)$ messages need to be delivered. The spanner topology is implicitly stored on the nodes with each node’s storage cost bounded by $O(\log \alpha)$. With such partial information stored at each node, there is a local distributed algorithm that finds a $(1+\varepsilon)$ -stretch path between any two nodes.

We remark that the idea of the greedy spanner resembles, on an abstract level, the ‘highway hierarchy’ in transportation networks. It has been shown that to find a shortest path to a destination, one only needs to search for a ‘highway entrance’ within certain radius, and search only on the highways beyond that. This turns out to substantially reduce the time to find shortest paths on such graphs [8, 36]. Our work provides a possible explanation of how the road system evolved to the way it is today. We propose to verify this in our future work.

Related work. In the vast amount of prior literature on geometric spanners, there are three main ideas: Θ -graphs, the greedy spanners, and the WSPD-induced spanners [33]. Please refer to the book for a nice survey [33]. We will review two spanner construction ideas that are most related to our approach. The first idea is the path-greedy spanner construction [13, 16–18]. All pairwise edges are ordered with non-decreasing lengths and checked in that order. An edge is included in the spanner if the shortest path in the current graph is longer than s times the Euclidean distance, and is discarded otherwise. Variants of this idea generate spanners with constant degree and total weight $O(|\text{MST}|)$. This idea cannot be applied in our setting as edges constructed in practice may not be in non-decreasing order of their lengths. The second idea is to use the gap property [13] — the sources and sinks of any two edges in an edge set are separated by a distance at least proportional to the length of the shorter of the two edges and their directions are differed no more than a given angle. The gap-greedy algorithm [7] considers pairs of points, again, in order of non-decreasing distances, and includes an edge in the spanner if and only if it does not violate the gap property. The spanner generated this way has constant degree and total weight $O(|\text{MST}|)$. Compared with our algorithm, our strategy is a relaxation of the gap property in the way that the edges in our spanner may have one of their endpoints arbitrarily close (or at the same points) and we have no restriction on the direction of the edges and the ordering of the edges to be considered. The proof for the gap greedy algorithm requires heavily plane geometry tools and our proof technique

only uses packing argument and can be extended to the general metric setting as long as a similar packing argument holds. To get these benefit our algorithm has slightly worse upper bounds on the spanner weight by a logarithmic factor.

Prior work on compact routing [37, 23, 3, 27, 2] usually implies a $(1 + \varepsilon)$ -spanner explicitly or implicitly. Again, these spanners are constructed in a coordinated setting.

2 Uncoordinated spanner construction algorithm

Given n points in \mathbb{R}^d , each point p will check whether an edge pq should be built. p builds pq only if there does not exist an edge $p'q'$ such that p and q are within distance $\frac{|p'q'|}{2(s+1)}$ from p', q' respectively.

This incremental construction of edges is executed by different agents in a completely uncoordinated manner. We assume that no two agents perform the above strategy at exactly the same time. Thus when any agent conducts the above process, the decision is based on the current network already constructed. The algorithm terminates when all agents finish checking the edges from themselves to all other points. We first examine the properties of the constructed graph G by these uncoordinated behaviors. We will discuss in Section 4 a proper complexity model for the uncoordinated construction in a distributed environment and also bound the computing cost of this spanner.

Before we proceed, we first realize the following invariant is maintained by the graph G . The proof follows immediately from the construction of G .

- Lemma 1.**
1. For any edge pq that is not in G , there is another edge $p'q'$ in G such that $|pp'| \leq |p'q'|/(2s+2)$, $|qq'| \leq |p'q'|/(2s+2)$.
 2. For any two edges $pq, p'q'$ in the constructed graph G , suppose that pq is built before $p'q'$, then one of the following is true: $|pp'| > |pq|/(2s+2)$ or $|qq'| > |pq|/(2s+2)$.

To show that the algorithm eventually outputs a good spanner, we first show the connection of G with the notion of *well-separated pair decomposition*.

Definition 1 (Well-separated pair). Let $t > 0$ be a constant, and a pair of sets of points A, B are well-separated with respect to t (or t -separated), if $d(A, B) \geq t \cdot \max(\text{diam}(A), \text{diam}(B))$, where $\text{diam}(A)$ is the diameter of the point set A , and $d(A, B) = \min_{p \in A, q \in B} |pq|$.

Definition 2 (Well-separated pair decomposition). Let $t > 0$ be a constant, and P be a point set. A well-separated pair decomposition (WSPD) with respect to t of P is a set of pairs $\mathcal{W} = \{\{A_1, B_1\}, \dots, \{A_m, B_m\}\}$, such that

1. $A_i, B_i \subseteq P$, and the pair sets A_i and B_i are t -separated for every i .
2. For any pair of points $p, q \in P$, there is at least one pair (A_i, B_i) such that $p \in A_i$ and $q \in B_i$.

Here m is called the size of the WSPD.

It is not hard to see that the uncoordinated spanner is equivalent to the following greedy algorithm that computes an s -WSPD.

1. Choose an arbitrary pair (p, q) , not yet covered by existing pairs in \mathcal{W} .
2. Include the pair of point sets $B_r(p)$ and $B_r(q)$ in the WSPD \mathcal{W} , with $r = |pq|/(2+2s)$, where $B_r(p)$ is the collection of points within distance r from point p .
3. Label the point pair (p_i, q_i) with $p_i \in B_r(p)$ and $q_i \in B_r(q)$ as being covered.
4. Repeat the above steps until every pair of points is covered.

Clearly the above algorithm produces a WSPD, as each pair $(B_r(p), B_r(q))$ is well-separated and all pairs of points are covered. The spanner edge (p, q) is one-to-one correspondence to the well-separated pair $(B_r(p), B_r(q))$ in the above algorithm — the rule in Lemma 1 prevented two edges from the same well-separated pair in \mathcal{W} to be constructed. Thus the number of edges in the spanner G is the same as the size of the greedy WSPD. It is already known that for any well-separated pair decomposition, if one edge is taken from each well-separated pair, then the edges will become a spanner on the original point set [6, 5, 32, 29]. For our specific greedy s -WSPD, we are able to get a slightly better stretch. The proof is omitted due to space constraint.

Theorem 1. *From the greedy s -WSPD, one build a graph G that includes each pair (p, q) when it is selected by the greedy algorithm. Thus G is a spanner with stretch factor $(s + 1)/(s - 1)$.*

To make the stretch factor as $1 + \varepsilon$, we just take $s = 1 + 2/\varepsilon$ in our spanner construction. Next, we show that the greedy WSPD algorithm will output a linear number of well-separated pairs.

3 A greedy algorithm for well-separated pair decomposition

We show the connection of the greedy WSPD with a specific WSPD constructed by the deformable spanner [22], in the way that at most a constant number of pairs in \mathcal{W} is mapped to each well-separated pair constructed by the deformable spanner. To be consistent, the greedy WSPD is denoted by \mathcal{W} and the WSPD constructed by the deformable spanner is denoted by \mathcal{W} .

Deformable spanner. Given a set of points P in the plane, a set of *discrete centers* with radius r is defined to be the maximal set $S \subseteq P$ that satisfies the *covering* property and the *separation* property: any point $p \in P$ is within distance r to some point $p' \in S$; and every two points in S are of distance at least r away from each other. In other words, all the points in P can be covered by balls with radius r , whose centers are exactly those points in the discrete center set S . And these balls do not cover other discrete centers.

We now define a hierarchy of discrete centers in an recursive way. S_0 is the original point set P . S_i is the discrete center set of S_{i-1} with radius 2^i . Without loss of generality we assume that the closest pair has distance 1 (as we can scale the point set and do not change the combinatorial structure of the discrete center hierarchy). Thus the number of levels of the discrete center hierarchy is $\log \alpha$, where α is the aspect ratio of the point set P , defined as the ratio of the maximum pairwise distance to the minimum pairwise distance, that is, $\alpha = \max_{u,v \in P} |uv| / \min_{u,v \in P} |uv|$. Since a point p may stay in multiple consecutive levels and correspond to multiple nodes in the discrete center hierarchy,

we denote by $p^{(i)}$ the existence of p at level i . For each point $p^{(i-1)} \in S_{i-1}$ on level i , it is within distance 2^i from at least one other point on level $i + 1$. Thus we assign to $p^{(i-1)}$ a *parent* $q^{(i)}$ in S_i such that $|p^{(i-1)}q^{(i)}| \leq 2^i$. When there are multiple points in S_i that cover $p^{(i-1)}$, we choose one as its parent arbitrarily. We denote by $P(p^{(i-1)})$ the parent of $p^{(i-1)}$ on level i . We denote by $P^{(i)}(p) = P(P^{(i-1)}(p))$ the *ancestor* of p at level i .

The deformable spanner is based on the hierarchy, with all edges between two points u and v in S_i if $|uv| \leq c \cdot 2^i$, where c is a constant equal to $4 + 16/\varepsilon$. We restate some important properties of the deformable spanner below.

Lemma 2 (Packing Lemma [22]). *In a point set $S \subseteq R^d$, if every two points are at least distance r away from each other, then there can be at most $(2R/r + 1)^d$ points in S within any ball with radius R .*

Lemma 3 (Deformable spanner properties [22]). *For a set of n points in R^d with aspect ration α ,*

1. *For any point $p \in S_0$, its ancestor $P^{(i)}(p) \in S_i$ is of distance at most 2^{i+1} away from p .*
2. *Any point $p \in S_i$ has at most $(1 + 2c)^d - 1$ edges with other points of S_i .*
3. *The deformable spanner \hat{G} is a $(1 + \varepsilon)$ -spanner G with $O(n/\varepsilon^d)$ edges.*
4. *\hat{G} has total weight $O(|MST| \cdot \lg \alpha / \varepsilon^{d+1})$, where $|MST|$ is the weight of the minimal spanning tree of the point set S .*

As shown in [22], the deformable spanner implies a well-separated pair decomposition $\hat{\mathcal{W}}$ by taking all the ‘cousin pairs’. Specifically, for a node $p^{(i)}$ on level i , we denote by P_i the collection of points that are descent of $p^{(i)}$ (including $p^{(i)}$ itself), called the *decendants*. Now we take the pair (P_i, Q_i) , the sets of decedents of a *cousin pair* $p^{(i)}$ and $q^{(i)}$, i.e., $p^{(i)}$ and $q^{(i)}$ are *not* neighbors in level i but their parents are neighbors in level $i + 1$. This collection of pairs constitutes a $\frac{4}{\varepsilon}$ -well-separated pair decomposition. The size of $\hat{\mathcal{W}}$ is bounded by the number of cousin pairs and is $O(n/\varepsilon^d)$.

Size of greedy WSPD. The basic idea is to map the pairs in the greedy WSPD \mathcal{W} to the pairs in $\hat{\mathcal{W}}$ and show that at most a constant number of pairs in \mathcal{W} map to the same pair in $\hat{\mathcal{W}}$.

Theorem 2. *The greedy s -WSPD \mathcal{W} has size $O(ns^d)$.*

Proof. Choose $c = 4(s + 1)$ (or, $s = c/4 - 1$) in the deformable spanner DS . The size of $\hat{\mathcal{W}}$ is $O(ns^d)$. Now we will construct a map from \mathcal{W} to $\hat{\mathcal{W}}$. Each pair $\{P, Q\}$ in \mathcal{W} is created by considering the points inside the balls $B_r(p), B_r(q)$ with radius $r = |pq|/(2 + 2s)$ around p, q . Now we consider the ancestors of p, q in the spanner DS respectively. There is a unique level i such that the ancestor $u_i = P^{(i)}(p)$ and $v_i = P^{(i)}(q)$ do not have a spanner edge in between but the ancestor $u_{i+1} = P^{(i+1)}(p)$ and $v_{i+1} = P^{(i+1)}(q)$ have an edge in between. The pair u_i, v_i is a cousin pair by definition and thus the decedents of them correspond to an s -well-separated pair in $\hat{\mathcal{W}}$. We say that the pair $(B_r(p), B_r(q)) \in \mathcal{W}$ maps to the descendant pair $(P_i, Q_i) \in \hat{\mathcal{W}}$.

By the discrete center hierarchy (Lemma 3), we show that,

$$|pq| \geq |u_i v_i| - |p u_i| - |q v_i| \geq |u_i v_i| - 2 \cdot 2^{i+1} \geq (c-4) \cdot 2^i.$$

The last inequality follows from that fact that u_i, v_i do not have an edge in the spanner and $|u_i v_i| > c \cdot 2^i$. On the other hand,

$$|pq| \leq |p u_{i+1}| + |u_{i+1} v_{i+1}| + |q v_{i+1}| \leq 2 \cdot 2^{i+2} + c \cdot 2^{i+1} = 2(c+4) \cdot 2^i.$$

The last inequality follows from the fact that u_{i+1}, v_{i+1} have an edge in the spanner and $|u_{i+1} v_{i+1}| \leq c \cdot 2^{i+1}$. Similarly, we have

$$c \cdot 2^i < |u_i v_i| \leq |u_i u_{i+1}| + |u_{i+1} v_{i+1}| + |v_i v_{i+1}| \leq 2 \cdot 2^{i+1} + c \cdot 2^{i+1} = 2(c+2) \cdot 2^i.$$

Therefore the distance between p and q is $c' \cdot |u_i v_i|$, where $(c-4)/(2c+4) \leq c' \leq (2c+8)/c$.

Now suppose two pair $(B_{r_1}(p_1), B_{r_1}(q_1)), (B_{r_2}(p_2), B_{r_2}(q_2))$ in \mathcal{W} map to the same pair u_i and v_i by the above process. Without loss of generality suppose that p_1, q_1 are selected before p_2, q_2 in our greedy algorithm. Here is the observation:

1. $|p_1 q_1| = c'_1 \cdot |u_i v_i|, |p_2 q_2| = c'_2 \cdot |u_i v_i|, r_1 = |p_1 q_1|/(2+2s) = c'_1 \cdot |u_i v_i|/(2+2s), r_2 = c'_2 \cdot |u_i v_i|/(2+2s)$, where $(c-4)/(2c+4) \leq c'_1, c'_2 \leq (2c+8)/c$, and r_1, r_2 are the radius of the balls for the two pairs respectively.
2. The reason that (p_2, q_2) can be selected in our greedy algorithm is that at least one of p_2 or q_2 is outside the balls $B(p_1), B(q_1)$, by Lemma 1. This says that at least one of p_2 or q_2 is of distance r_1 away from p_1, q_1 .

Now we look at all the pairs (p_ℓ, q_ℓ) that are mapped to the same ancestor pair (u_i, v_i) . The pairs are ordered in the same order as they are constructed, i.e., p_1, q_1 is the first pair selected in the greedy WSPD algorithm. Suppose r_{min} is the minimum among all radius r_i . $r_{min} \geq c/(2c+8) \cdot |u_i v_i|/(2+2s) = |u_i v_i|/(4s+8)$. We group these pairs in the following way. The first group H_1 contains (p_1, q_1) and all the pairs (p_ℓ, q_ℓ) that have p_ℓ within distance $r_{min}/2$ from p_1 . We say that (p_1, q_1) is the representative pair in H_1 and the other pairs in H_1 are *close* to the pair (p_1, q_1) . The second group H_2 contains, among all remaining pairs, the pair that was selected in the greedy algorithm the earliest, and all the pairs that are close to it. We repeat this process to group all the pairs into k groups, H_1, H_2, \dots, H_k . For all the pairs in each group H_j , we have one representative pair, denoted by (p_j, q_j) and the rest of the pairs in this group are close to it.

We first bound the number of pairs belonging to each group by a constant with a pack argument. With our group criteria and the above observations, all p_ℓ in the group H_j are within radius r_{min} away from each other. This means that the q_ℓ 's must be far away — the q_ℓ 's must be at least distance r_{min} away from each other, by Lemma 1. On the other hand, all the q_ℓ 's are descendant of the node v_i , so $|v_i q_\ell| \leq 2^{i+1}$ by Theorem 3. That is, all the q_ℓ 's are within a ball of radius 2^{i+1} centered at v_i . By the packing Lemma 2, the number of such q_ℓ 's is at most $(2 \cdot 2^{i+1}/r_{min} + 1)^d \leq (2 \cdot 2^{i+1}(4s+8)/|u_i v_i| + 1)^d \leq (4(s+2)/(s+1) + 1)^d$. This is also the bound on the number of pairs inside each group.

Now we bound the number of different groups, i.e., the value k . For the representative pairs of the k groups, $(p_1, q_1), (p_2, q_2), \dots, (p_k, q_k)$, all the p_i 's must be at least distance $r_{min}/2$ away from each other. Again these p_i 's are all descendant of u_i and thus are within distance 2^{i+1} from u_i . By a similar packing argument, the number of such p_i 's is bounded by $(4 \cdot 2^{i+1}/r_{min} + 1)^d \leq (8(s+2)/(s+1) + 1)^d$. So the total number of pairs mapped to the same ancestor pair in \mathcal{W} will be at most $(4(s+2)/(s+1) + 1)^d \cdot (8(s+2)/(s+1) + 1)^d = (O(1 + 1/s))^d$. Thus the total number of pairs in W is at most $O(ns^d)$. This finishes the proof.

With the connection of the greedy WSPD with the uncoordinated spanner construction in Section 2, we immediately get the following theorem (with proofs omitted).

Theorem 3. *The uncoordinated spanner with parameter s is a spanner with stretch factor $(s+1)/(s-1)$ and has $O(ns^d)$ number of edges, a maximal degree of $O(\lg \alpha \cdot s^d)$, average degree $O(s^d)$, and total weight $O(\lg \alpha \cdot |MST| \cdot s^{d+1})$.*

4 Spanner construction and applications

The uncoordinated spanner construction can be applied for peer-to-peer system design, to allow users to maintain a spanner in a distributed manner. For that, we will first extend our spanner results to a metric with constant doubling dimension. The doubling dimension of a metric space (X, d) is the smallest value γ such that each ball of radius R can be covered by at most 2^γ balls of radius $R/2$ [25].

Theorem 4. *For n points and a metric space defined on them with constant doubling dimension γ , the uncoordinated spanner construction outputs a spanner G with stretch factor $(s+1)/(s-1)$, has total weight $O(\gamma^2 \cdot \lg \alpha \cdot |MST| \cdot s^{O(\gamma)})$ and has $O(\gamma^2 \cdot n \cdot s^{O(\gamma)})$ number of edges. Also it has a maximal degree of $O(\gamma \cdot \lg \alpha \cdot s^{O(\gamma)})$ and average degree $O(\gamma \cdot s^{O(\gamma)})$.*

Distributed construction. Now we would like to discuss the model of computing for P2P overlay design as well as the construction cost of the uncoordinated spanner. We assume that there is already a mechanism maintained in the system such that any node x can obtain the distance to any node y in $O(1)$ time. For example, this can be done by a TRACEROUTE command executed by x to the node y . We also assume that there is a service answering near neighbor queries: given a node p and a distance r , return the neighbors within distance r from p . Such an oracle is often maintained in a distributed file sharing system. Various structured P2P system support such function with low cost [30]. Even in unstructured system such as BitTorrent, the Ono plugin is effective at locating nearby peers, with vanishingly small overheads [1].

The spanner edges are recorded in a distributed fashion so that no node has the entire picture of the spanner topology. After each edge pq is constructed, the peers p, q will inform their neighboring nodes (those in $B_r(p)$ and $B_r(q)$ with $r = |pq|/(2s+2)$) that such an edge pq exists so that they will not try to connect to one another. We assume that these messages are delivered immediately so that when any newly built edge is informed to nodes of relevance. The number of messages for this operation is

bounded by $|B_r(p)| + |B_r(q)|$. The amount of storage at each node x is proportional to the number of well-separated pairs that include x . The following theorem bounds the total number of such messages during the execution of the algorithm and the amount of storage at each node.

Theorem 5. *For the uncoordinated spanner G and the corresponding greedy WSPD $\mathcal{W} = \{(P_i, Q_i)\}$ with size m , each node x is included in at most $O(s^d \lg \alpha)$ well-separated pairs in \mathcal{W} . Thus, $\sum_{i=1}^m (|P_i| + |Q_i|) = O(ns^d \cdot \lg \alpha)$.*

Distributed routing. Although the spanner topology is implicitly stored on the nodes with each node only knows some piece of it, we are actually able to do a distributed and local routing on the spanner with only information available at the nodes such that the path discovered has maximum stretch $(s + 1)/(s - 1)$. In particular, for any node p who has a message to send to node q , it is guaranteed that (p, q) is covered by a well-separated pair $(B_r(p'), B_r(q'))$ with $p \in B_r(p')$ and $q \in B_r(q')$. By the construction algorithm, the edge $p'q'$, after constructed, is informed to all nodes in $B_r(p') \cup B_r(q')$, including p . Thus p includes in the packet a partial route with $\{p \rightsquigarrow p', p'q', q' \rightsquigarrow q\}$. The notation $p \rightsquigarrow p'$ means that p will need to first find out the low-stretch path from p to the node p' (inductively), from where the edge $p'q'$ can be taken, such that with another low-stretch path to be found out from q' to q , the message can be delivered to q . This way of routing with partial routing information stored with the packet is similar to the idea of source routing [38] except that we do not include the full routing path at the source node. By the same induction as used in the proof of spanner stretch (Theorem 1), the final path is going to have stretch at most $(s + 1)/(s - 1)$.

Nearest neighbor search. We remark that with the spanner each node can easily find its nearest neighbor. Recall that each point x keeps all the pairs (p, q) that create a ‘dumb-bell’ pair set covering x . Then we claim, among all these p , one of them must be the nearest neighbor of x . Otherwise, suppose y is the nearest neighbor of x , and y is not one of p . But in the WSPD, (x, y) will belong to one of the pair set (P_i, Q_i) , which correspond to a pair (p', q') . Then there is a contradiction, as $|xp'| < |xy|$ implies that y is not the nearest neighbor of x . Thus one’s nearest neighbor is locally stored at this node already. According to Theorem 5, x will belong to at most $O(s^d \lg \alpha)$ different pair sets. So the nearest neighbor search can be finished in $O(s^d \lg \alpha)$ time by using just the local information.

References

1. <http://www.aqualab.cs.northwestern.edu/projects/Ono.html>.
2. I. Abraham, C. Gavoille, A. V. Goldberg, and D. Malkhi. Routing in networks with low doubling dimension. In *Proc. of the 26th International Conference on Distributed Computing Systems (ICDCS)*, July 2006.
3. I. Abraham and D. Malkhi. Compact routing on euclidian metrics. In *PODC '04: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 141–149, New York, NY, USA, 2004. ACM.
4. S. Albers, S. Eilts, E. Even-Dar, Y. Mansour, and L. Roditty. On nash equilibria for a network creation game. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 89–98, New York, NY, USA, 2006. ACM.

5. S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. Smid. Euclidean spanners: short, thin, and lanky. In *Proc. 27th ACM Symposium on Theory Computing*, pages 489–498, 1995.
6. S. Arya, D. M. Mount, and M. Smid. Randomized and deterministic algorithms for geometric spanners of small diameter. In *Proc. 35th IEEE Symposium on Foundations of Computer Science*, pages 703–712, 1994.
7. S. Arya and M. Smid. Efficient construction of a bounded-degree spanner with low weight. *Algorithmica*, 17:33–54, 1997.
8. H. Bast, S. Funke, D. Matijevic, P. Sanders, and D. Schultes. In transit to constant time shortest-path queries in road networks. In D. Applegate and G. Brodal, editors, *9th Workshop on Algorithm Engineering and Experiments (ALENEX'07)*, pages 46–59, New Orleans, USA, 2007. SIAM.
9. Callahan and Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In *Proc. 4th ACM-SIAM Symposium on Discrete Algorithms*, pages 291–300, 1993.
10. P. B. Callahan. Optimal parallel all-nearest-neighbors using the well-separated pair decomposition. In *Proc. 34th IEEE Symposium on Foundations of Computer Science*, pages 332–340, 1993.
11. P. B. Callahan and S. R. Kosaraju. Algorithms for dynamic closest-pair and n -body potential fields. In *Proc. 6th ACM-SIAM Symposium on Discrete Algorithms*, pages 263–272, 1995.
12. P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k -nearest-neighbors and n -body potential fields. *J. ACM*, 42:67–90, 1995.
13. B. Chandra, G. Das, G. Narasimhan, and J. Soares. New sparseness results on graph spanners. *Internat. J. Comput. Geom. Appl.*, 5:125–144, 1995.
14. Y. Chu, S. Rao, S. Seshan, and H. Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. *SIGCOMM Comput. Commun. Rev.*, 31(4):55–67, 2001.
15. J. Corbo and D. Parkes. The price of selfish behavior in bilateral network formation. In *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 99–107, New York, NY, USA, 2005. ACM.
16. G. Das, P. Heffernan, and G. Narasimhan. Optimally sparse spanners in 3-dimensional Euclidean space. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 53–62, 1993.
17. G. Das and G. Narasimhan. A fast algorithm for constructing sparse Euclidean spanners. *Internat. J. Comput. Geom. Appl.*, 7:297–315, 1997.
18. G. Das, G. Narasimhan, and J. Salowe. A new way to weigh malnourished Euclidean graphs. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms*, pages 215–222, 1995.
19. D. Eppstein. Spanning trees and spanners. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 425–461. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.
20. J. Erickson. Dense point sets have sparse Delaunay triangulations. In *Proc. 13th ACM-SIAM Symposium on Discrete Algorithms*, pages 125–134, 2002.
21. A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. On a network creation game. In *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 347–351, 2003.
22. J. Gao, L. Guibas, and A. Nguyen. Deformable spanners and their applications. *Computational Geometry: Theory and Applications*, 35(1-2):2–19, 2006.
23. L.-A. Gottlieb and L. Roditty. Improved algorithms for fully dynamic geometric spanners and geometric routing. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 591–600, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

24. J. Gudmundsson, C. Levcopoulos, G. Narasimhan, and M. Smid. Approximate distance oracles for geometric graphs. In *Proc. 13th ACM-SIAM Symposium on Discrete Algorithms*, pages 828–837, 2002.
25. A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *FOCS '03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 534–543, 2003.
26. T. Jansen and M. Theile. Stability in the self-organized evolution of networks. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 931–938, New York, NY, USA, 2007. ACM.
27. G. Konjevod, A. W. Richa, and D. Xia. Optimal-stretch name-independent compact routing in doubling metrics. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 198–207, 2006.
28. M. Kwon and S. Fahmy. Topology-aware overlay networks for group communication. In *NOSSDAV '02: Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video*, pages 127–136, New York, NY, USA, 2002. ACM.
29. C. Levcopoulos, G. Narasimhan, and M. H. M. Smid. Improved algorithms for constructing fault-tolerant spanners. *Algorithmica*, 32(1):144–156, 2002.
30. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys & Tutorials, IEEE*, pages 72–93, 2005.
31. T. Moscibroda, S. Schmid, and R. Wattenhofer. On the topologies formed by selfish peers. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 133–142, New York, NY, USA, 2006. ACM.
32. G. Narasimhan and M. Smid. Approximating the stretch factor of Euclidean graphs. *SIAM J. Comput.*, 30:978–989, 2000.
33. G. Narasimhan and M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007.
34. D. Peleg. *Distributed Computing: A Locality Sensitive Approach*. Monographs on Discrete Mathematics and Applications. SIAM, 2000.
35. S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Topologically-aware overlay construction and server selection. In *Proceedings of the 21th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, volume 3, pages 1190–1199, 2002.
36. P. Sanders and D. Schultes. Engineering highway hierarchies. In *ESA'06: Proceedings of the 14th conference on Annual European Symposium*, pages 804–816, London, UK, 2006. Springer-Verlag.
37. A. Slivkins. Distance estimation and object location via rings of neighbors. In *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 41–50, 2005.
38. A. S. Tanenbaum. *Computer networks (3rd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.
39. W. Wang, C. Jin, and S. Jamin. Network overlay construction under limited end-to-end reachability. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, volume 3, pages 2124–2134, March 2005.
40. X. Zhang, Z. Li, and Y. Wang. A distributed topology-aware overlays construction algorithm. In *MG '08: Proceedings of the 15th ACM Mardi Gras conference*, pages 1–6, New York, NY, USA, 2008. ACM.