Testing Euclidean Spanners

Frank Hellweg, Melanie Schmidt, and Christian Sohler*

Department of Computer Science, Technische Universität Dortmund

Abstract. We develop a *property testing algorithm* with query complexity $\tilde{\mathcal{O}}(\delta^{-5d}\epsilon^{-5}D\log^6 \varDelta\sqrt{n})$ that tests whether a directed geometric graph $G = (P, E)$ with maximum degree D and vertex set $P \subseteq \{1, \ldots, \varDelta\}^d$ (for constant d) is a Euclidean $(1 + \delta)$-spanner. Such a property testing algorithm accepts every $(1 + \delta)$-spanner and rejects with high constant probability every graph that is ϵ-far from this property, i. e., every graph that differs in more than $\epsilon|P|$ edges from every $(1 + \delta)$-spanner.

1 Introduction

Property testing is the computational task of deciding whether a given object has a predetermined property Π or is far away from every object with property Π. Thus, property testing can be viewed as a relaxation of a standard decision problem. The main goal of property testing is to develop randomized algorithms that perform this relaxed decision task by only looking at a small part of the input object, i.e. we want to develop algorithms whose running time is sublinear in the object's description size. Property testing has been introduced by Rubinfeld and Sudan [34] and the study of combinatorial properties has been initiated by Goldreich, Goldwasser, and Ron [26]. Since then, property testing algorithms have been developed for properties of functions [25,24,10], properties of distributions [8,7], algebraic properties [11,34,29], graph and hypergraph properties [26,3,16,9], and *geometric properties* which we continue to study in this paper. Previous work on geometric property testing includes testing algorithms for convexity of polygons [21], convexity [33], geometric properties of point sets (for example convex position), the Euclidean minimum spanning tree [18,17,19], and clusterability of point sets [1,17]. The area of sublinear time algorithms is closely related to property testing; geometric properties studied in this field include polyhedron intersection and point location in planar convex subdivisions with bounded face size [13] and approximation algorithms for the weight of the Euclidean minimum spanning tree [15] and metric minimum spanning trees [20].

Euclidean spanners. In this paper, we consider Euclidean spanners. A weighted directed geometric graph $G = (P, E)$ is a directed graph whose vertex set is a set of points in the Euclidean space $\mathbb{R}^d$ and whose edge weights (lengths) are given by the Euclidean distance of the vertices. Throughout this paper, we will assume that d is constant. A geometric graph is called a (Euclidean) $(1+\delta)$-*spanner*, if for

* Research partly supported by DFG grant SO 514/3-1.

every pair of vertices p and q the shortest path distance $d_G(p,q)$ in G is at most $(1+\delta)\cdot\|p-q\|_2$. Euclidean spanners are a fundamental geometric graph structure as they can be used to approximately solve many geometric proximity problems. Many different constructions of Euclidean spanners are known. For constant δ, Euclidian $(1+\delta)$-spanners with a linear number of edges can for example be constructed by using so-called Θ-graphs [14,28] or structures based on the well-separated pair decomposition [12,31]. Techniques to construct spanners with bounded degree are also known [5]. For more details we refer to the book [31]. We investigate the question whether a given graph is a Euclidean spanner. The related question of computing the stretch factor $(1+\delta)$ of a given graph has recently been studied in [4,22,30]. Additionally, Ahn et al. [2] discuss the problem to find an edge whose removal leads to the smallest possible increase in the stretch factor, and Farshi et al. [23] consider the question which edge should be added to receive the best decrease in the stretch factor (both articles consider very special cases only).

Our contribution We develop a property testing algorithm that distinguishes spanners from geometric graphs that are very different from being a spanner. The distance of a given graph to a spanner is measured by the amount of edges that have to be inserted to establish the spanner property, and graphs with a high distance to every spanner are called ϵ-far (where 'high' is defined depending on the parameter ϵ). A property tester is a sublinear randomized algorithm that has to reject every ϵ-far graph with high probability. We develop a property tester with one-sided error, i. e., every spanner is always accepted (where in the general case, this must only happen with high probability). The analysis of our algorithm assumes that the vertices of the input graph lie on a d-dimensional discrete grid $\{1,\ldots,\Delta\}^d$. The query complexity and running time of our algorithm is $\tilde{\mathcal{O}}(\delta^{-5d}\epsilon^{-5}D\log^6\Delta\sqrt{n})$, so it depends logarithmically on the range of possible coordinates. It is open whether this influence or the exponential dependence on the dimension can be avoided.

2 Preliminaries

Let $G=(P,E)$ be a directed geometric graph with point set $P:=\{p_1,\ldots,p_n\}\subseteq\mathbb{R}^d$, d constant, and edge set $E\subseteq P\times P$. We will assume that the algorithm is given the number of vertices n, but it does not know the positions of the points in P. It may query the position of the i-th vertex in $O(1)$ time for any index i, $1\le i\le n$. The graph structure is stored in the adjacency list model for general (sparse) graphs proposed in [32]. In this model, it is possible to query the degree $\deg(p_i)$ of the i-th vertex in $O(1)$ time by specifying the index i, $1\le i\le n$ and one can obtain the j-th edge incident to the i-th vertex in $O(1)$ time for any $1\le i\le n$ and $1\le j\le\deg(i)$. The maximum degree of a vertex in G is denoted by D. The query complexity of a property testing algorithm is the number of queries of coordinates, degrees and neighbors that it needs to accomplish its task.

We use $[p, q\rangle$ to denote a directed edge from p to q and denote the shortest path distance from p to q by $d_G(p, q)$, i. e., $d_G(p, p) = 0$, $d_G(p, q) = ||p-q||_2$ for all $p, q \in P$ with $[p, q\rangle \in E$ and $d_G(p, q) := \min_{\text{paths } Q \text{ from } p \text{ to } q} \sum_{[p',q'\rangle \in Q} d_G(p', q')$ for all other pairs of points. A directed geometric graph is a $(1 + \delta)$-spanner if it provides a $(1 + \delta$-approximation of the Euclidean distances between all pairs of points as stated in the following definition. Note that we restrict to $(1 + \delta)$-spanners with $0 < \delta < 1$.

Definition 1. *Let* $0 < \delta < 1$ *be given. A directed geometric graph* G *is called a* $(1+\delta)$-spanner*, if* $d_G(p, q) \leq (1+\delta)||q-p||_2$ *for all pairs of points* $[p, q\rangle \in P \times P$ *with* $p \neq q$.

The definition of property testers depends on the definition of ϵ-farness. We define that a graph is ϵ-far from being a spanner if one has to add, delete or modify more than ϵn edges of the graph to get a spanner. This is slightly different from the definition in [32] where $\epsilon|E|$ is used instead of ϵn.

Definition 2. *Let* $G = (P, E)$ *be a directed geometric graph and let* $0 < \epsilon < 1$ *and* $0 < \delta < 1$ *be given.* G *is* ϵ-far *from a directed geometric graph* $G' = (P, E')$ *if* $|E \backslash E' \cup E' \backslash E| > \epsilon n$. G *is* ϵ-far *from having the property to be a* $(1+\delta)$*-spanner if it is* ϵ*-far from every* $(1 + \delta)$*-spanner.*

We say that an algorithm $\mathcal{A}$ is a *property tester with one-sided error* for the property of being a $(1 + \delta)$-spanner if it returns

– *true* with a probability of 1 if G is a $(1 + \delta)$-spanner
– *false* with probability at least 2/3 if G is ϵ-far from being a $(1 + \delta)$-spanner

when given input $|V|$, ϵ and δ and access to a directed geometric graph $G = (P, E)$ as described above.

3 The Algorithm

Our algorithm for testing geometric spanners depends on two parameters s and s' that are specified at the end of this paragraph. It has access to a directed geometric graph in the way described above. First, the algorithm samples a set of s vertices $q_1, \ldots, q_s$ uniformly at random. Then it independently performs a shortest path computation starting at each vertex q_i. Dijkstra's algorithm is used until s' vertices have been reached (in the current iteration). We denote this traversal as *Dijkstra traversal.* After all Dijkstra traversals are finished, the algorithm checks whether a certificate that the graph is not a spanner was found. Let W_i be the maximum graph distance reached during the Dijkstra traversal started at q_i. Then we know that all points within graph distance W_i were reached. Thus, if a point lying within distance $W_i/(1+\delta)$ of q_i was sampled but not reached, then the graph is rejected because in order to satisfy the spanner property for such a point, there has to be a path that is completely included in the searched part of the graph. Thus, the graph cannot be a spanner. If no such point is found for any q_i then the graph is accepted. We consider two versions of our algorithm. For the analysis in Section 4 we set $s' := \tilde{\mathcal{O}}(\delta^{-2d}\epsilon^{-1})$, for the general case discussed in Section 5 we set $s' := \tilde{\mathcal{O}}(\delta^{-4d}\epsilon^{-3}\log^6 \Delta)$, where the points are assumed to come from the discrete d-dimensional space $\{1,\ldots,\Delta\}^d$. In both cases, the parameter s is set such that $s := \tilde{\mathcal{O}}(\delta^{-d}\epsilon^{-2}\sqrt{n})$.

The maximum degree of a vertex in the graph influences the running time and query complexity of the Dijkstra traversals, because to perform such a computation one needs to collect information about all neighbors of the already searched vertices. If the maximum degree is constant, this is, of course, possible in constant time for each vertex.

```
UniformTester($n, G, \delta, \epsilon$)
    Sample $s$ points $q_1, \ldots, q_s$ from $P$ without replacement
    for $i \leftarrow 1$ to $s$
        Perform a Dijkstra traversal in $G$ from $q_i$ until
            $s'$ nodes have been visited
        Let $R$ be the set of vertices visited and let $W = \max_{q \in R} d_G(q_i, q)$
        forall sample points $q_j$ such that $\|q_i - q_j\|_2 < \frac{1}{1+\delta} \cdot W$
            if $q_j \notin R$ or $d_G(q_i, q_j) \geq (1+\delta)\|q_i - q_j\|_2$
                return false
    return true
```

Note that this algorithm only tests small neighborhoods of certain points. At first glance it seems unlikely that the spanner property can be tested by local investigations. Consider a path whose vertices are placed on the boundary of an ellipsoid. Then δ can be chosen in a way such that the spanner property is satisfied for all pairs of points except the one with highest distance. This means that the violation cannot be found by sampling only parts of the path. Surprisingly, when distinguishing spanners and graphs that are ϵ-far from being a spanner, the situation is different: A graph that locally satisfies the spanner property can be transformed into a spanner by adding only a few edges. We show that a geometric graph cannot be ϵ-far from being a spanner if it does only contain 'global' violations of the spanner property.

4 Testing Graphs with Uniformly Spread Vertices

In this section we show that for input graphs of a certain kind the algorithm UniformTester is a property tester with one-sided error, i. e., it accepts every spanner with probability one and rejects every graph far from being a spanner with high probability. The first property holds as the algorithm only rejects a graph if it has found a pair of points that violates the spanner property, so it remains to show the high rejection probability. The special case that we assume is that the points of G are uniformly spread over the plane in the following sense.

Definition 3. *Let G be a directed geometric graph with n points and let n_u be a value depending on n (and $1/\delta$ and $1/\epsilon$). We say that G is* uniformly n_u-distributed on grid H *if there exists a d-dimensional grid H with a width of $S = \sqrt[d]{n/n_u}$ cells in each dimension such that every cell of H contains $\mathcal{O}(n_u)$ points of G and such that H completely contains G. We denote the cells of H as* base cells *and their side length by w_0.*

Now let G be a uniformly n_u-distributed graph on grid H. We say that a pair of points (p, q) is a *violating pair* if $d_G(p, q) > (1 + \delta)||p - q||_2$. If G does not contain a violating pair, then it is a spanner, and if G is ϵ-far from being a spanner, there have to be at least ϵn violating pairs (otherwise adding less than ϵn edges would establish the spanner property). For our algorithm, we need that (some of) these violating pairs can be found by exploring small neighborhoods of certain points. For this purpose we show that if the spanner property holds for all 'close' pairs of points in G, then it can be established for distant pairs of points by inserting no more than $\epsilon n/2$ edges into G. This means that if G is ϵ-far from being a spanner, there have to be at least $\epsilon n/2$ 'close' pairs of points violating the spanner property. To realize this, we use exponential grids that are constructed in a similar manner as in [27] where they are used to compute coresets for clustering problems. The basic idea behind this is to connect clouds of points with a small number of edges which is somewhat similar to the use of well-seperated pair decompositions for spanner constructions [12,31]. However, our focus lies on connecting 'distant' points with few edges and not on computing a complete spanner.

For our exponential grid it is convenient to triple the side length of the boxes, so we wish to deal with powers of three and for simplicity assume that the side length S is a power of three.

Definition 4. *Let G be a uniformly n_u-distributed graph on grid H and let c be a cell of H. We set $i_\delta := \min\{i \mid 3^i \geq (8\sqrt{d}/\delta), i \in \mathbb{N}\}$ and define $B_{c,i}$ for $i \geq 0$ as the cube that is centered at the center of c and has side length $3^{i+i_\delta} \cdot w_0$. Let $p \in P$. We denote the cell of H containing p by $c(p)$. Then we define the* geometric neighborhood *$\Gamma(p)$ of p as set of all points that are contained in the area inside $B_{c(p),0}$. A point $q \in P$ is a* neighbor of *p if $q \in \Gamma(p)$. Additionally we define the* extended geometric neighborhood *$\Gamma^e(p)$ of p as the set of all points that are contained in the area inside $B_{c(p),1}$ (note that $\Gamma(p) \subseteq \Gamma^e(p)$).*

To construct the exponential grid H_c around c, we define $w_i = 3^{i-1} \cdot w_0$, cover $B_{c,i}$ for $i \geq 1$ with $[(3^{i+i_\delta} \cdot w_0)/3^{i-1} \cdot w_0]^d = (3^{i_\delta+1})^d$ cubes of side length w_i and include a cube into H_c if it is not contained in a smaller box $B_{c,j}, j < i$ but has a non-empty intersection with H (i. e., we cut off cubes that lie aside of G). Note that H_c does not contain cells lying within $B_{c,0}$.

Observe that the number of cells of H that lie in $\Gamma(p)$ is at most $(3^{i_\delta})^d < (3 \cdot 8\sqrt{d}/\delta)^d = O(1/\delta^d)$ for all c in H (it may be less if c lies close to the margin of H). Likewise, the number of cells of H that lie in $\Gamma^e(p)$ is $\mathcal{O}(1/\delta^d)$. For the number of cubes in all exponential grids we state the following. Due to space limitations, the proof of this statement and of the other statements in this section are omitted.

Observation 1. *The value n_u can be chosen such that $n_u = \mathcal{O}(\delta^{-d}\epsilon^{-1} \log \delta n)$ and such that for every n_u-distributed graph on grid H the total number of cubes in all exponential grids H_c for all $c \in H$ is less than $\epsilon n/2$.*

Now we use the structure defined by H to insert $\epsilon n/2$ edges into G that ensure that any two points p, $q \in P$ with $q \notin \Gamma(p)$ are connected by a path of length $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$ if this is already true for all pairs consisting of a point and one of its neighbors. This works because the exponential grid grows in such a manner that an arbitrary connection between a cell c and a cell in the exponential grid H_c always suffices to establish the spanner property for all points in these cells, if applied recursively.

Lemma 1. *Let G be a uniformly n_u-distributed graph on grid H which satisfies $d_G(p,q) \leq (1+\delta)||p-q||_2$ for all pairs consisting of a point $p \in P$ and a neighbor $q \in \Gamma(p)$ of p. Then G is $\epsilon n/2$-close to being a spanner.*

Lemma 1 implies that our algorithm only has to test whether there are at least $\epsilon n/2$ violating pairs consisting of a point $p \in P$ and a neighbor $q \in \Gamma(p)$ of p. The next Lemma gives an upper bound on the number of points that have to be explored for a given p to find all neighbors q' of p such that p and q' do *not* form a violating pair, i. e., for all neighbors q of p that are not found during such a search, p and q form a violating pair. This is basically due to the definition of the neighborhoods.

Lemma 2. *Let G be a uniformly n_u-distributed graph on grid H and let $p \in P$ be a point. It suffices to explore $\mathcal{O}(\delta^{-d} n_u)$ points of G to find all neighbors $q \in \Gamma(p)$ with $d_G(p,q) \leq (1+\delta)||p-q||_2$.*

Lemma 2 states how many points in the neighborhood have to be explored for each sampled point to test whether a point belongs to a sampled violating pair. Finally we have to ensure that the algorithm samples enough points to get both points of at least one violating pair with high probability.

Lemma 3. *Let $G = (P, E)$ be a directed geometric graph. Assume that it holds that every point $p \in P$ has at most ρ neighbors and that every point $q \in P$ is neighbor of at most ρ points. Let there be $\epsilon n/2$ pairs of a point p and a neighbor $q \in \Gamma(p)$ such that p and q form a violating pair. Then there is an $s = \mathcal{O}(\rho \cdot \sqrt{n} \cdot \epsilon^{-1})$ such that the probability that s points sampled uniformly at random contain both points of one of these violating pairs is at least* $5/6$.

The proof of Lemma 3 is a standard analysis similar to the birthday problem.

Now we combine the statements to get the main result of this section. Inserting the value of n_u given in Observation 1 into Lemma 2 shows that exploring $\mathcal{O}\left(\delta^{-d} \cdot \frac{1}{\epsilon} \cdot \log(\delta n) \cdot \delta^{-d}\right) = \tilde{\mathcal{O}}(\delta^{-2d}\epsilon^{-1})$ points suffices to ensure that the algorithm only rejects if it has indeed found a violating pair. By Lemma 3 we gain that sampling $\mathcal{O}(\delta^{-d}\epsilon^{-2} \log \delta n \sqrt{n}) = \tilde{\mathcal{O}}(\delta^{-d}\epsilon^{-2}\sqrt{n})$ points is sufficient to find both points of a violating pair with high probability. Combining both facts we get that algorithm UniformTester is a property tester as defined in Section 2 and that it has a query complexity of $\tilde{\mathcal{O}}\left(\delta^{-3d}\epsilon^{-3}D\sqrt{n}\right)$. When implemented using adequate data structures, the query complexity and running time of the algorithm only differs by logarithmic factors.

Theorem 1. *The algorithm* UniformTester *is a property tester for the property of being a* $(1+\delta)$*-spanner under the assumption that the input graphs are uniformly* n_u*-distributed on grid* H *with* $n_u = \mathcal{O}(\delta^{-d}\epsilon^{-1}\log \delta n)$ *chosen as in Observation 1. It has a query complexity and running time of* $\tilde{\mathcal{O}}\left(\delta^{-3d}\epsilon^{-3}D\sqrt{n}\right)$.

5 Testing Graphs with Vertices Placed on a Discrete Grid

From now on, we assume that the vertices of G are coming from the discrete d-dimensional space $\{1,\ldots,\Delta\}^d$. We show that under this assumption after adapting the number of vertices to explore for each sampled point the above algorithm is a property tester for the property of G being a $(1+\delta)$-spanner with a query complexity of $\tilde{\mathcal{O}}(\delta^{-5d}\epsilon^{-5}D\log^6\Delta\sqrt{n})$. As above, we state that the algorithm only rejects the input if it finds a witness not satisfying the $(1+\delta)$-spanner property. This ensures that an input graph cannot be rejected if it is a $(1+\delta)$-spanner. Thus, it remains to show that for all input graphs that are ϵ-far from being $(1+\delta)$-spanners we find such a witness with high probability. The proof idea is based on the previous proof, but in this case instead of a grid, H will be defined as a (d-dimensional) quadtree partitioning, satisfying that no leaf cell of the quadtree exceeds a certain number of vertices; i. e., we build the quadtree by starting with a bounding box including all points and partitioning each box into 2^d subboxes, if it contains more than s points for some $s \in \mathcal{O}\left(\text{poly}(\log\Delta, \frac{1}{\epsilon}, \frac{1}{\delta})\right)$ recursively.

Similar to the uniform case, we construct an exponential grid around every leaf box c of H and insert an edge from c to every cell of its exponential grid. By bounding the depth of H, we can bound the number of leaf boxes in H and thus show that the number of edges we insert in this way does not exceed $\epsilon n/4$. The difference to the uniform case is that the neighborhood of a leaf box c may contain many smaller leaf boxes and thus arbitrarily many points of P. A neighborhood whose number of points exceeds a certain amount cannot be completely explored by our algorithm; this means, that for the proof we have to ensure that one can fix any violation of the spanner property occuring in such neighbourhoods by inserting at most $\epsilon n/4$ edges in total. We show that there are only few such neighborhoods, which implies that we can establish the $(1+\delta)$-spanner property for these cells by inserting few edges.

We define the exponential grid around a leaf box of H similarly to Section 4:

Definition 5. *Let* $G = (P, E)$ *be a directed geometric graph whose points come from the d-dimensional grid* $T = \{1,\ldots,\Delta\}^d$ *and let* H *be a d-dimensional quadtree of the points of* G*, satisfying that no leaf box of* H *contains more than*

$s = \frac{(3^{i_\delta})^d \cdot 2^{d+3} \cdot \log^2 2\Delta}{\epsilon \log 2} = \mathcal{O}(\frac{\log^2 \Delta}{\delta^d \epsilon})$ *points of* G. *For* $p \in P$, *we denote the leaf box of* H *that contains* p *by* $c(p)$. *Let* c *be a leaf box of* H. *We define* $w(c)$ *as the side length of* c *and* $B_{c,i}$ *for* $i \geq 0$ *as the cube that is centered at the center of* c *and has side length* $3^{i+i_\delta} \cdot w(c)$. *We call the points that are contained in the area inside* $B_{c,0}$ *its* geometric neighborhood $\Gamma(c) \subseteq P$ *and for two points* $p, q \in P$ *we call* p geometric neighbor *of* q *if* p *lies in the geometric neighborhood of* q.

Define $w_i(c) = 3^{i-1} \cdot w(c)$, $i > 0$. *To construct the exponential grid* H_c *around* c, *cover* $B_{c,i}$ *for* $i \geq 1$ *with* $[(3^{i+i_\delta} \cdot w(c))/3^i \cdot w(c)]^d = (3^{i_\delta})^d$ *cubes of side length* $w_i(c)$ *and include a cube into* H_c *if it is not contained in a smaller box* $B_{c,j}, j < i$ *but it is contained in* H.

Note that since $\delta < 1$, the *extended neighborhood* $\hat{\Gamma}(c) := B_{c,1} \cap P$ of c contains all points which can be part of a path from a point $p \in c$ to a point $q \in \Gamma(c)$ having a length of at most $(1+\delta)||p-q||_2$.

Definition 6. *We call a leaf box* $c \in H$ samplable *if* $\hat{\Gamma}(c)$ *contains at most* $k := \frac{2^{d+3} \cdot 3^{d(2i_\delta+1)} s \log^2 \Delta}{\epsilon} = \mathcal{O}(\frac{\log^4 \Delta}{\delta^{3d} \epsilon^2})$ *cells* $c_1, \ldots c_k$ *with* $w(c_i) < w(c)$.

We start the analysis by bounding the total number of cells in the exponential grids around the leaf boxes by $\frac{\epsilon n}{4}$. The depth of the quadtree defining H is bounded by $\log \Delta$, since the grid T has a side length of Δ and the smallest possible box has one of $\sqrt[d]{s}$. Since a box must contain at least $s+1$ vertices to be split into subboxes, the overall number of leaf boxes is at most $\frac{2^d n \log \Delta}{s}$. This leads to the following Lemma:

Lemma 4. *The following statements hold:*

- *The number of cells in all exponential grids around leaf boxes of* H *does not exceed* $\epsilon n/8$.
- *There are at most* $\frac{\epsilon n}{8 \cdot 3^{d i_\delta} s^2}$ *cells that are not samplable.*

Proof. The first statement follows by bounding the maximum number of cells in each of the exponential grids and multiplying with the above bound on the number of quadtree leaf boxes. For the second statement consider an arbitrary leaf box c. What is the number of larger leaf boxes c' such that c is contained in $\hat{\Gamma}(c')$? For each possible size of c' – there are at most $\log \Delta$ possible sizes – due to the number of equal-sized boxes contained in an extended neighbourhood there are at most $(3^{i_\delta+1})^d \log \Delta$ such leaf boxes. Since there are at most $\frac{2^d n \log \Delta}{s}$ leaf boxes, the total number of *is-contained*-relations is at most $\frac{2^d \cdot 3^{d(i_\delta+1)} n \log^2 \Delta}{s}$. Thus, there are at most

$$\frac{2^d \cdot 3^{d(i_\delta+1)} n \log^2 \Delta}{sk} = \frac{\epsilon n}{8 \cdot 3^{d i_\delta} s^2}$$

cells that are not samplable. □

Lemma 5. *Let* G *be* ϵ*-far from being a* $(1+\delta)$*-spanner. Then there are at least* $\epsilon n/2$ *violating pairs of points in the geometric neighborhoods of samplable leaf boxes of* H.

Proof. Assume that there are less than $\epsilon n/2$ violating points in the geometric neighborhoods of samplable leaf boxes of H. Then we can insert edges into G as follows:

– Insert edges from every leaf box $c \in H$ to all cells of its exponential grid and vice versa; let E_1 be the set of these edges. Due to Lemma 4 it holds that $|E_1| < \epsilon n/4$.
– For every leaf box $c \in H$ that is not samplable, insert edges $[p,q\rangle$ and $[q,p\rangle$ for all $p \in c, q \in \Gamma(c)\backslash(\{p\} \cup X_c)$, where $X_c = \{\hat{c} \in H : \hat{c} \cap \Gamma(c) \neq \emptyset \wedge w(\hat{c}) < w(c)\}$, i. e., excluding those points of $\Gamma(c)$ that lie in leaf boxes of H that are smaller than c. Let E_2 be the set of these edges. Due to Lemma 4 there are at most $\frac{\epsilon n}{8 \cdot 3^{di_\delta} s^2}$ leaf boxes that we treat in this way. Since we insert at most $2 \cdot 3^{di_\delta} s^2$ edges for each of them, we insert in total at most $\epsilon n/4$ edges.
– Insert edges between all violating pairs of nodes in geometric neighborhoods of leaf boxes $c \in H$ that are samplable; let E_3 be the set of these edges. Due to our assumption, $|E_3| \leq \epsilon n/2$.

Let $G' = (P, E \cup E_1 \cup E_2 \cup E_3)$ be the resulting graph. We have inserted at most ϵn edges into G to obtain G'.

Claim. G' is a $(1+\delta)$-spanner.

Proof. Let $p, q \in P$ be arbitrary points. We show $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$ by induction over $\pi = ||p-q||_2$ (note that since all points of G' lie on a Δ^d-grid, the number of possible distances is finite). At first let $||p-q||_2 = m := \min_{p',q' \in P} ||p'-q'||_2$. Since this is the smallest possible distance, it holds $p \in \Gamma(c(q))$ and $q \in \Gamma(c(p))$. We consider the following cases:

1. $c(p)$ is samplable. Since the insertion of E_3 ensures that there are no violating pairs of points inside $\Gamma(c(p))$, we have $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$.
2. $c(p)$ is not samplable and $w(c(p)) \leq w(c(q))$. Since $[p,q\rangle \in E_2$, it holds $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$.
3. $c(p)$ is not samplable and $w(c(p)) > w(c(q))$. We consider two subcases:
   (a) $c(q)$ is samplable. Thus, the insertion of E_3 ensures that there are no violating pairs of points inside $\Gamma(c(p))$, ensuring that $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$.
   (b) $c(q)$ is not samplable. In this case, it holds $[p,q\rangle \in E_2$ and therefore $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$.

For the inductive step let $||p-q||_2 = \pi$; the induction hypothesis is that $d_{G'}(p',q') \leq (1+\delta)||p'-q'||_2$ holds for all $p', q' \in P$ such that $||p'-q'||_2 < \pi$. We consider the following cases:

1. $p \in \Gamma(c(q))$ and $q \in \Gamma(c(p))$. This case follows analogously to the case $||p-q||_2 = m$.
2. $p \notin \Gamma(c(q))$, but $q \in \Gamma(c(p))$.
   (a) $c(p)$ is samplable. In this case the insertion of E_3 ensures that $d_{G'}(p,q) \leq (1+\delta)||p-q||_2$.

(b) $c(p)$ is not samplable. If $w(c(p)) \leq w(c(q))$, then $[p, q\rangle \in E_2$. If $w(c(p)) > w(c(q))$, then p lies in a cell $c \in H_{c(q)}$ since it is not contained in $\Gamma(c(q))$. Therefore E_1 contains an edge $[r_1, r_2\rangle$, where $r_1 \in c(q)$ and $r_2 \in c$. Due to the construction of $H_c(q)$, it holds $||r_2 - p||_2 < ||p - q||_2 = \pi$ and $||r_1 - q||_2 < ||p - q||_2 = \pi$. Thus, by applying the induction hypothesis we get $d_{G'}(r_2, p) \leq (1 + \delta)||r_2 - p||_2$ and $d_{G'}(r_1, q) \leq (1 + \delta)||r_1 - q||_2$. Then $d_{G'}(p, q) \leq (1 + \delta)||p - q||_2$ can be proved as in Lemma 1.

3. $q \notin \Gamma(c(p))$. This case is analogous to the case that $w(c(p)) < w(c(q))$ in 2(b) except that we consider $H_{c(p)}$ instead of $H_{c(q)}$. □

We finish the proof of Lemma 5 by stating that Claim 5 is a contradiction to the assumption that G is ϵ-far from being a $(1 + \delta)$-spanner. □

Now reconsider our Algorithm. The largest number of points that can be contained in the extended neighborhood of some samplable leaf box $c \in H$ is $(k + (3^{i_\delta + 1})^d)s = \tilde{\mathcal{O}}(\delta^{-4d}\epsilon^{-3} \log^6 \Delta)$. By setting the number of vertices to explore by a Dijkstra traversal to this number, we can ensure that the extended geometric neighborhood of any sample point p_i is completely explored, if $c(p_i)$ is samplable. Since there are at least $\epsilon n/2$ violating pairs of points in such neighborhoods, we can apply Lemma 3 and state the following theorem.

Theorem 2. *Let $G = (P, E)$ be a geometric graph with $P \subseteq \{1, \ldots, \Delta\}^d$. Then there is a property testing algorithm with query complexity and running time $\tilde{\mathcal{O}}(\delta^{-5d}\epsilon^{-5}D \log^6 \Delta\sqrt{n})$ that accepts G, if G is a $(1 + \delta)$-spanner and rejects G with probability at least $2/3$, if G is ϵ-far from being a $(1 + \delta)$-spanner.*

6 A Lower Bound for Testing Spanners

We give a sketch of the proof of a lower bound of $\Omega(n^{\frac{1}{3}})$ for the number of queries of property testing algorithms with one-sided error. Our main idea is that it is not possible to distinguish between a line of $2k$ subsequent points and two coinciding lines of k subsequent points with only one-sided error and $o(n^{\frac{1}{3}})$ queries. By permutating the vertices, one can define two classes of graphs such that a tester with one-sided error cannot decide to which of the classes a randomly chosen graph belongs. But all graphs of the first type are 1-spanners, and all graphs of the second type are not spanners (regardless of the stretch factor) as the two lines are not connected at all. They are also ϵ-far from being a spanner as there are k pairs of points with zero distance that have to be connected individually. The following Theorem states the result. Its proof is similiar to the proof of Theorem 4.2 in [6].

Theorem 3. *Any property tester for the property of being a $(1+\delta)$-spanner with one-sided error has a query complexity of $\Omega(n^{\frac{1}{3}})$.*

Note that the construction can be modified such that there are no coinciding points by randomly moving the positions of the points to the left or to the right for the first type of graphs and moving one point of each pair of coinciding points to the left and the other to the right for the second type of graphs.

Acknowledgements

We thank Petra Berenbrink, Oded Goldreich and Ilan Newman for helpful discussions, anonymous referees for helpful comments and Mohammad Ali Abam for pointing out reference [30].

References

1. Alon, N., Dar, S., Parnas, M., Ron, D.: Testing of Clustering. SIAM Journal on Discrete Mathematics 16(3), 393–417 (2003)
2. Ahn, H.-K., Farshi, M., Knauer, C., Smid, M., Wang, Y.: Dilation-Optimal Edge Deletion in Polygonal Cycles. In: Algorthims and Computation, pp. 88–99. Springer, Heidelberg (2007)
3. Alon, N., Fischer, E., Newman, I., Shapira, A.: A combinatorial characterization of the testable graph properties: it's all about regularity. SIAM Journal on Computing 39(1), 143–167 (2009)
4. Agarwal, P.K., Klein, R., Knauer, C., Langerman, S., Morin, P., Sharir, M., Soss, M.: Computing the Detour and Spanning Ratio of Paths, Trees, and Cycles in 2D and 3D. Discr. & Computational Geometry 39(1-3), 17–37 (2007)
5. Arya, S., Das, G., Mount, M., Salowe, J.S., Smid, M.: Euclidean spanners: short, thin, and lanky. In: Proceedings of the 27th Annnual ACM Symposium on the Theory of Computing (STOC), pp. 489–498 (1995)
6. Ben-Zwi, O., Lachish, O., Newman, I.: Lower bounds for testing Euclidean Minimum Spanning Trees. Information Processing Letters 102(6), 219–225 (2007)
7. Batu, T., Fortnow, L., Fischer, E., Kumar, R., Rubinfeld, R., White, P.: Testing Random Variables for Independence and Identity. In: Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS), pp. 442–451 (2001)
8. Batu, T., Fortnow, L., Rubinfeld, R., Smith, W., White, P.: Testing that distributions are close. In: Proceedings of the 41st IEEE Symposium on Foundations of Computer Science (FOCS), pp. 259–269 (2000)
9. Benjamini, I., Schramm, O., Shapira, A.: Every minor-closed property of sparse graphs is testable. In: Proceedings of the 40th Annnual ACM Symposium on the Theory of Computing (STOC), pp. 393–402 (2008)
10. Blais, E.: Testing juntas nearly optimally. In: Proceedings of the 41st Annnual ACM Symposium on the Theory of Computing (STOC), pp. 151–158 (2009)
11. Blum, M., Luby, M., Rubinfeld, R.: Self-Testing/Correcting with Applications to Numerical Problems. In: Proceedings of the 22nd Annnual ACM Symposium on the Theory of Computing (STOC), pp. 73–83 (1990)
12. Callahan, P.B., Kosaraju, S.R.: Faster algorithms for some geometric graph problems in higher dimensions. In: Proceedings of the 4th Annnual ACM-SIAM Symposium on Discr. Algorithms (SODA), pp. 291–300 (1993)
13. Chazelle, B., Liu, D., Magen, A.: Sublinear Geometric Algorithms. SIAM Journalon Computing 35(3), 627–646 (2006)
14. Clarkson, K.L.: Approximating algorithms for shortest path motion planning. In: Proceedings of the 19th Annnual ACM Symposium on the Theory of Computing (STOC), pp. 56–65 (1987)
15. C̃zumaj, A., Ẽrgün, F., F̃ortnow, L., M̃agen, A., Ñewman, I., R̃ubinfeld, R., S̃ohler, C.: Approximating the Weight of the Minimum Spanning Tree in Sublinear Time. SIAM Journal on Comp. 35(1), 91–109 (2005)

16. Czumaj, A., Shapira, A., Sohler, C.: Testing hereditary properties of nonexpanding bounded-degree graphs. SIAM Journal on Computing 38(6), 2499–2510 (2009)
17. Czumaj, A., Sohler, C.: Property Testing with Geometric Queries. In: Meyer auf der Heide, F. (ed.) ESA 2001. LNCS, vol. 2161, pp. 266–277. Springer, Heidelberg (2001)
18. Czumaj, A., Sohler, C., Ziegler, M.: Property Testing in Computational Geometry. In: Paterson, M. (ed.) ESA 2000. LNCS, vol. 1879, pp. 155–166. Springer, Heidelberg (2000)
19. Czumaj, A., Sohler, C.: Testing Euclidean minimum spanning trees in the plane. ACM Transactions on Alg. 4(3) (2008)
20. Czumaj, A., Sohler, C.: Estimating the Weight of Metric Minimum Spanning Trees in Sublinear Time. SIAM Journal on Computing 39(3), 904–922 (2009)
21. Ergun, F., Kannan, S., Kumar, R., Rubinfeld, R., Viswanathan, M.: Spot-Checkers. J. of Computer and System Sciences 60(3), 717–751 (2000)
22. Eppstein, D., Wortman, K.A.: Minimum dilation stars. Computational Geometry: Theory and Applications 37(1), 27–37 (2007)
23. Farshi, M., Giannopoulos, P., Gudmundsson, J.: Finding the best shortcut in a geometric network. In: Proceedings of the 21th Annnual ACM Symposium on Computational Geometry, pp. 327–335 (2005)
24. Fischer, E., Lehman, E., Newman, I., Raskhodnikova, S., Rubinfeld, R., Samorodnitsky, A.: Monotonicity testing over general poset domains. In: Proceedings of the 34th Annnual ACM Symposium on the Theory of Computing (STOC), pp. 474–483 (2002)
25. Goldreich, O., Goldwasser, S., Lehman, E., Ron, D., Samorodnitsky, A.: Testing Monotonicity. Combinatorica 20(3), 301–337 (2000)
26. Goldreich, O., Goldwasser, S., Ron, D.: Property Testing and its Connection to Learning and Approximation. J. of the ACM 45(4), 653–750 (1998)
27. Har-Peled, S., Mazumdar, S.: On Coresets for k-Means and k-Median Clustering. In: Proceedings of the 36th Annnual ACM Symposium on the Theory of Computing (STOC), pp. 291–300 (2004)
28. Keil, M.: Approximating the complete Euclidean graph. In: Karlsson, R., Lingas, A. (eds.) SWAT 1988. LNCS, vol. 318, pp. 208–213. Springer, Heidelberg (1988)
29. Kaufman, T., Sudan, M.: Algebraic property testing: the role of invariance. In: Proceedings of the 40th Annnual ACM Symposium on the Theory of Computing (STOC), pp. 403–412 (2008)
30. Narasimhan, G., Smid, M.: Approximating the Stretch Factor of Euclidean Graphs. SIAM Journal on Computing, 978–989 (2000)
31. Narasimhan, G., Smid, M.: Geometric Spanner Networks. Cambridge University Press, Cambridge (2007)
32. Ron, D., Parnas, M.: Testing the Diameter of Graphs. Random Structures & Algorithms 20(2), 165–183 (2002)
33. Rademacher, L., Vempala, S.: Testing Geometric Convexity. In: Proceedings of the 24th Foundations of Software Technology and Theoretical Computer Science (FSTTCS), pp. 469–480 (2004)
34. Rubinfeld, R., Sudan, M.: Robust Characterizations of Polynomials with Applications to Program Testing. SIAM Journal on Computing 25(2), 252–271 (1996)