SIAM J. Discrete Math.
Vol. 27, No. 4, pp. 1759–1769

PROXIMITY ALGORITHMS FOR NEARLY DOUBLING SPACES[*]

LEE-AD GOTTLIEB[†] AND ROBERT KRAUTHGAMER[‡]

Abstract. We introduce a new problem in the study of doubling spaces: Given a point set S and a target dimension d^*, remove from S the fewest number of points so that the remaining set has doubling dimension at most d^*. We present a bicriteria approximation for this problem and extend this algorithm to solve a group of proximity problems.

Key words. doubling dimension, metric spanners, approximate minimum spanning tree, approximate distance oracle, all points nearest neighbor

AMS subject classifications. 68W25, 46B85, 05C12

DOI. 10.1137/120874242

1. Introduction.

In the last few years, researchers have increasingly made use of the doubling dimension in the design of algorithms. Analyzing algorithmic tasks via the doubling dimension is natural for proximity problems such as nearest neighbor search [33, 9, 15] and clustering [38, 4, 18] and for graph problems such as spanner construction [19, 12, 17, 24, 25], the traveling salesman problem [38, 6], and routing [30, 37, 3, 32, 31]. The doubling dimension has proved to be a powerful tool in embeddings [5, 26, 1, 2, 11, 8, 23] and has found applications in fields such as machine learning [10, 21]. Interestingly, the problem of computing the exact doubling dimension of a point set is NP-hard. (This result seems to be folklore.) Yet this fact has not deterred the development of algorithms that are based on the doubling dimension, partly because it can be approximated within a constant factor, and partly because many of these algorithms function without explicit knowledge of the doubling dimension—it appears only in the analysis.

However, a host of algorithms previously developed for spaces of bounded doubling dimension—perhaps even the majority of them—suffer from a more serious problem: They are not robust to severe yet infrequent irregularities in the space. The guarantees provided by these algorithms are markedly degraded even if only a small subset of the working set possesses high doubling dimension. This problem was noted, for example, by [13], who instead defined a global notion of dimension (which can be thought of as the average doubling dimension over the set) and developed an algorithm under this new definition.

We pursue a different approach. We introduce the following key problem: Given an n-point set S and a target dimension d^*, remove from S the fewest number of points so that the remaining set has doubling dimension at most d^* (or equivalently, target doubling constant $\lambda^* = 2^{d^*}$). We thus call a data set *nearly doubling* if all but a negligible fraction of the points have bounded doubling dimension.

A solution to this point removal problem yields a contribution in two related areas. The first paradigm, broadly speaking, is outlier detection. In this scenario, the

[*] Received by the editors April 19, 2012; accepted for publication (in revised form) August 12, 2013; published electronically October 16, 2013. A preliminary version of this paper appeared in *Proceedings of APPROX* 2010. This work was supported in part by the Israel Science Foundation (grant 452/08) and by a Minerva grant.

http://www.siam.org/journals/sidma/27-4/87424.html

[†] Department of Computer Science, Ariel University, Ariel, Israel (leead@ariel.ac.il).

[‡] Weizmann Institute of Science, Rehovot, Israel (robert.krauthgamer@weizmann.ac.il).

removed points are ignored and only the remaining points are processed. A direct motivation for this model stems from the dimension induced clustering framework of [20], which given a point set seeks a subset with low intrinsic dimension. Further motivation stems from algorithms which have "slack"; that is, they give guarantees for most but not all of the point set [32, 18]. These algorithm can be extended to nearly doubling data sets by simply ignoring the removed points (i.e., throwing them into the slack). The second paradigm is an original one: Here, both the removed points and the remaining ones are processed, albeit by separate algorithms tailored to the properties of the two point sets.

Results. The point removal problem is NP-hard, and it is not difficult to show that the problem does not admit even an approximate multiplicative-factor solution (see Lemma 2.1). However, we develop a framework that yields a bicriteria approximation for this problem. In section 3, we present algorithms achieving the bounds listed below, where we assume the algorithms have access to the point set and to an oracle that answers in $O(1)$ time a query asking for the distance between two points.

1. In time $2^{O(d^*)}n^4$, the algorithm removes a number of points arbitrarily close to optimal while obtaining doubling dimension $4d^* + O(1)$ (Corollary 3.6).
2. In time $2^{O(d^*)}n \log \alpha$ (where α is the aspect ratio of S), the algorithm removes a number of points arbitrarily close to optimal while obtaining doubling dimension $10d^* + O(1)$ (Corollary 3.8).
3. In time $2^{O(d^*)}n \log^3 n$, the algorithm removes a number of points arbitrarily close to optimal while obtaining doubling dimension $12d^* + O(1)$ (Corollary 3.8).

Returning to the first motivating paradigm presented above, our algorithms solve the clustering problem posed by [20]. (They provided heuristic solutions to this question.) In section 4, we present algorithms that function under the second paradigm delineated above—these algorithms process the removed points and the remaining ones using separate techniques tailored to the properties of the two point sets. When the data set is nearly doubling, or more precisely, when all but at most the square root of the number of points have bounded doubling dimension, we give near-linear time algorithms for constructing $(1+\varepsilon)$-stretch spanners, approximate minimum spanning trees (MSTs), $O(1)$-query time distance oracles, and calculating approximate all points nearest neighbor (i.e., calculating an approximate nearest neighbor for each point of the set).

2. Preliminaries.

In this section we define doubling dimension and present some basic hardness results. We then review point hierarchies for doubling spaces.

Doubling dimension. For a metric (X, d), let λ be the infimum value such that every closed ball in X can be covered by λ closed balls of half the radius, where a ball is centered at a point of the metric. λ is the *doubling constant* of X, and the *doubling dimension* of X is $\dim(X) = \log_2 \lambda$. A metric is *doubling* when its doubling dimension is finite. It is a folklore result that determining the doubling constant (and dimension) of a point set is an NP-hard problem. We formalize this result below.

LEMMA 2.1. *Given a metric (S, d), computing the doubling constant of S is NP-hard.*

Proof. The proof is a reduction from DOMINATING SET with bounded degree Δ, which is known to be NP-hard [34, 14]. Let $G = (V, E)$ be an input instance of DOMINATING SET with degree $\Delta < \sqrt{|V|} - 1$. Note that the size of any dominating set of G must then be greater than $\sqrt{|V|}$. Create a set S containing $|V|$ points, each corresponding to a vertex in V. Let $d(u, v) = \frac{1}{2}$ for $u, v \in S$ if the corresponding vertices have an edge in E, and let $d(u, v) = 1$ otherwise. The radius of S is 1.

Now, any subset of S found in a closed ball of radius $\frac{1}{2}$ contains at most $\Delta + 1 < \sqrt{|V|}$ points, so the doubling constant of such a subset is less than $\sqrt{|V|}$. However, a minimum cover of the entire set S by closed balls of radius $\frac{1}{2}$ is equivalent to a minimum dominating set of V, which in particular is greater than $\sqrt{|V|}$. It follows that the doubling constant of S is determined by covering all of S and is thus equivalent to computing a minimum dominating set of V. □

Notice that the above reduction preserves hardness of approximation and that it allows a degree bound that is not a constant, as long as $\Delta < \sqrt{|V|} - 1$. Approximating DOMINATING SET within a factor of $\Omega(\ln |V|)$ in graphs of maximum degree less than $\sqrt{|V|} - 1$ is NP-hard. Indeed, the hardness of DOMINATING SET follows from a simple reductions from SET COVER (see, e.g., [14]) and known hardness of approximation results for the latter [36]. And our degree bound can be imposed via a simple self-reduction: take the input graph instance, and replace it by a disjoint union of $|V| + 1$ copies of the instance, to obtain a graph on $V' = |V|^2 + |V|$ vertices and maximum degree $|V| - 1 < \sqrt{|V'|} - 1$. It follows that the inapproximability factor for our bounded-degree instances of DOMINATING SET remains proportional to $\ln |V| = \Omega(\ln |V'|)$. Altogether, we conclude that it is NP-hard to approximate the doubling constant of an n-point metric within a factor of $\Omega(\ln n)$. This problem does admit a weak form of approximation—for example, a 2-approximation to the doubling *dimension* (equivalently, the square of the doubling constant) can be determined by the algorithm of Lemma 3.3 (see also [27, Theorem 9.1]).

A further consequence of Lemma 2.1 is that the problem of removing the minimum number of points from a set S in order to obtain a set S' with some target doubling constant does not admit a multiplicative-factor approximation algorithm: That is, it is NP-hard to distinguish the case where no points need be removed from the case that one point must be removed.

Point hierarchies. Here, we define point hierarchies and describe three different hierarchies that have appeared in the literature and will be utilized in this paper.

Similar to what was described in [19, 33], a subset of points $X \subseteq Y$ is an (r, s)-discrete center set (or *net* in the terminology of [33]) of Y $(r \leq s)$ if it satisfies the following properties:

(i) Packing: For every $x, y \in X$, $d(x, y) \geq r$.

(ii) Covering: Every point $y \in Y$ is strictly within distance s of some point $x \in X$: $d(x, y) < s$.

We say that x *covers* y if $x \in X$, $y \in Y$ and $d(x, y) \leq s$. The previous conditions require that the points of X be spaced out yet nevertheless cover all points of Y. A hierarchy for a set S is a series of discrete center sets, where each level of the hierarchy is a discrete center set of the level beneath it. The bottom level contains all points, and the top level contains only a single point. For ease of presentation, we assume throughout that the minimum interpoint distance in S is 1.

The first hierarchy we describe is that of [33]. The hierarchy is composed of levels H_{2^i} (for integer $i = 0, \ldots$), where each level H_{2^i} $(i > 0)$ is a $(2^i, 2^i)$-discrete center set for the previous level $H_{2^{i-1}}$. (The subscript in the notation of the level indicates that the packing and covering properties of subsequent levels grow by a factor of 2.) The bottom level of the hierarchy is the set $Y_{2^0=1} = S$, and the top level is the set $Y_{2^{\lceil \log \alpha \rceil}}$ that contains only a single point. The construction supports insertions and deletions to the hierarchy—and its subsequent repair—in time $2^{O(\log \lambda)} \log \alpha$. (Recall that α is the aspect ratio of S.)

The second hierarchy is that of [24]. This hierarchy is similar to that of [33], but level H_{2^i} is a $(\frac{1}{2}2^i, 2^i)$-discrete center set for $H_{2^{i-1}}$. This hierarchy supports insertions and deletions in $2^{O(\log \lambda)} \log^3 n$ amortized time, and so a series of n insertions and deletions can be done deterministically in $2^{O(\log \lambda)} n \log^3 n$ time.

The third hierarchy is that of [15]. In this hierarchy, level H_{5^i} is a $(\frac{1}{5}5^i, \frac{3}{5}5^i)$-discrete center set for $H_{5^{i-1}}$. (The packing and covering properties of subsequent levels grow by a factor of 5.) The hierarchy supports insertions in time $2^{O(\log \lambda)} \log n$, though points cannot be removed from within the hierarchy. A static hierarchy with similar construction time was also presented in [28].

On top of these hierarchies, we define a parent-child relationship: Point $y \in H_{2^i}$ (or H_{5^i}) is the child of one of the points in $H_{2^{i+1}}$ (or $H_{5^{i+1}}$) that covers y. This immediately defines an ancestral relationship as well.

3. Point removal algorithm.

In this section, we present the bicriteria algorithm for the problem of removing points to obtain a target doubling constant.[1] The construction, presented below, proceeds roughly as follows: We formulate the notion of a "bad" witness set, which can be found efficiently and exists if and only if the doubling constant is too large (to within some constant factors). Given this setup, the algorithm is greedy: Repeatedly find such a witness set and remove it entirely.

We first define the density constant (in section 3.1) and explain the existence of witness sets for the density constant. We show that it is NP-hard to locate a maximum witness set, but we are able to give an approximation algorithm for locating witness sets. In section 3.2, we use this approximation algorithm for witness sets to develop a bicriteria point removal algorithm for achieving a target density constant. This bicriteria algorithm in turn yields a bicriteria point removal algorithm for achieving a target doubling constant. Finally, in section 3.3, we show how to improve the runtime of the two bicriteria algorithms.

3.1. Density constant and witness sets.

Let a closed ball $B(x, r) \in S$ be centered at point x and include all points of set S within distance r of x. We define the *density constant* $\mu(S)$ of point set S as follows: $\mu(S)$ is the smallest number such that every open r-radius ball of S (for every r) contains at most $\mu(S)$ points of mutual interpoint distance greater than $r/2$. Clearly the doubling constant cannot be greater than the density constant. Further, the density constant is not greater than the square of the doubling constant (since $\mu(S)$ balls of radius $\frac{r}{4}$ are required to cover these points). It follows that

$$\sqrt{\mu(S)} \leq \lambda(S) \leq \mu(S).$$

Now, we consider the following point removal problem: Given a point set S and a target density constant $\mu^* \leq \mu(S)$, remove the minimum number of points from S to obtain a set S^* with density constant μ^*. (This problem can serve as a proxy for the problem of removing points to obtain a target doubling constant.) However, we demonstrate in Lemma 3.1 below that the problem of determining the density constant of a point set S is NP-hard. An immediate consequence of Lemma 3.1 is that the point removal problem to achieve a target density constant is NP-hard.

LEMMA 3.1. *Given a point set S, the problem of determining the density constant of S is NP-hard.*

[1] One could alternatively consider a different goal of achieving a target doubling constant by altering a minimum number of *interpoint distances* (rather than points). Similar problems have been considered in the context of property testing; see, for example, [35].

Proof. The proof is a reduction from the MAXIMUM INDEPENDENT SET PROBLEM with bounded degree Δ [34]. Let $G = (V, E)$ be an input instance of the MAXIMUM INDEPENDENT SET PROBLEM with degree $\Delta < \sqrt{|V|}$. Note that the size of any maximal independent set for G is greater than $\sqrt{|V|}$. Create a set S containing $|V|$ points, each corresponding to a vertex in V. Let $d(u, v) = \frac{1}{2}$ for $u, v \in S$ (and an infinitely small ε) if the corresponding vertices have an edge in E, and let $d(u, v) = 1$ otherwise. The radius of S is 1.

Now, any subset of S found in a closed ball of radius $\frac{1}{2}$ contains fewer than $\sqrt{|V|}$ points (since the degree of V is less than $\sqrt{|V|}$), so the density constant of any subset of points of S that all fall in a ball of radius $\frac{1}{2}$ and have interpoint distance greater than $\frac{1}{4}$ is less than $\sqrt{|V|}$. However, the maximum number of points in all of S with interpoint distance greater than $\frac{1}{2}$ is necessarily greater than $\sqrt{|V|}$. It follows that determining the density constant of S is equivalent to determining the maximum independent set in V. ◻

Note that the reduction preserves hardness of approximation: It is NP-hard to approximate the density constant of a point set S within a factor of $|S|^{\frac{1}{2}-\varepsilon}$ (this follows easily from [29]).

It follows from Lemma 3.1 that the point removal problem to achieve a target density constant is NP-hard. Further, this problem does not even admit a multiplicative-factor approximation algorithm: It is NP-hard to distinguish the case where no points need be removed from the case where one point must be removed. However, we can approximate the density constant of a point set, as in Lemma 3.3 below. We will first require a definition.

DEFINITION 3.2. *Given a point set S, a* witness set *$S' \subset S$ is a set of points contained in a closed ball of radius r with mutual interpoint distance greater than $\frac{r}{2}$.*

Comment. The existence of a witness set $S' \subset S$ implies that $\mu(S) \geq |S'|$. The notion of a witness set exists for the density constant, but a similar notion does not exist for the doubling constant. That is, the addition of points to a set S with doubling constant $\lambda(S)$ may in fact result in a set with somewhat lower doubling constant than $\lambda(S)$. For example, the metric space defined by the four vectors $(1, 1), (1, -1), (-1, 1), (-1, -1)$ equipped with ℓ_2 distances has doubling constant four, since the points can be covered by a ball of radius $2\sqrt{2}$ centered at any of these points, while the minimum interpoint distance in the set is 2. Yet the metric space defined by the same vectors along with $(1, 0), (-1, 0)$ has doubling constant three, since the new points can serve as centers of smaller balls each covering three points. However, adding new points can decrease the doubling dimension by at most a constant factor, because in a set of interpoint distance at least r, each point must be found in a different $\frac{r}{4}$-radius ball, irrespective of the addition of new points. This disparity underlies our decision to define the density constant.

LEMMA 3.3. *Given an n-point set S with minimum interpoint distance* 1*, there exists an $O(2^{O(\log \mu(S))} n^3)$ time algorithm that locates a witness set of size $\lceil \sqrt{\mu(S)} \rceil$.*

Proof. Note that there are $O(n^2)$ interpoint distances in S, so there exist $O(n^2)$ distinct balls of S, and of course each is of size $O(n)$. For each ball $B(x \in S, r)$, we greedily build the point hierarchy of [33] consisting of three radii levels $\{r, \frac{r}{2}, \frac{r}{4}\}$, where level r contains only one point. This can be done in time $2^{O(\log \mu)} n$ per ball (where $\mu = \mu(S)$), yielding a total runtime of $O(2^{O(\log \mu)} n^3)$.

Now there must exist in S a point set S' of size exactly μ with radius r and minimum interpoint distance greater than $r/2$ for some r. In the hierarchy for the ball that contains S' (and possibly contains other points as well), exactly one of the following must hold:

(i) Level $\frac{r}{2}$ contains at least $\lceil\sqrt{\mu}\rceil$ points; it follows that these points are contained in a ball of radius r and have minimum interpoint distance greater than $\frac{r}{2}$, so that they are a witness set. Or,

(ii) Level $\frac{r}{2}$ contains fewer than $\lceil\sqrt{\mu}\rceil$ points. Now, since the μ points of S' have minimum distance $\frac{r}{2}$, they must be covered by distinct points of level $\frac{r}{4}$, so there must exist more than μ points in level $\frac{r}{4}$. It follows that some point of level $\frac{r}{2}$ covers more than $\sqrt{\mu}$ points of level $\frac{r}{4}$. These points have minimum interpoint distance greater than $\frac{r}{4}$ and are found in a set of radius less than $\frac{r}{2}$, so they are a witness set. $\square$

Comment. The algorithm of Lemma 3.3 yields a 2-approximation to the doubling dimension of S.

Lemma 3.3 shows that the density constant can be approximated. In the next section, we will use this tool to develop a bicriteria algorithm for the problem of removing points to obtain a target density constant. This will in turn allow us to develop a bicriteria algorithm for the problem of removing points to obtain a target doubling constant. However, for the purposes of efficient algorithmic runtime, we need to introduce a slightly stronger variant of Lemma 3.3, as follows.

LEMMA 3.4. *Given an n-point set S with minimum interpoint distance* 1 *and a parameter $\mu' \leq \mu(S)$, there exists an $2^{O(\log \mu')}n^4$ time algorithm that locates a maximal collection of distinct witness sets each of size $\lceil\sqrt{\mu'}\rceil$.*

Proof. The construction is similar to the one presented in the proof of Lemma 3.3. We identify all $O(n^2)$ balls, and for each ball $B(x,r)$ we build the top three levels of its hierarchy. The hierarchy of [33] for constant levels supports insertions and deletions in time $2^{O(\log \mu')}$, so a single hierarchy can be constructed in time $2^{O(\log \mu')}n$, and all $O(n^2)$ hierarchies in time $2^{O(\log \mu')}n^3$.

If during the construction of the three-level hierarchy for $B(x,r)$, we find that a newly inserted point implies a witness set of size $\lceil\sqrt{\mu'}\rceil$—that is, either level $\frac{r}{2}$ contains $\lceil\sqrt{\mu'}\rceil$ points or a point of level $\frac{r}{2}$ covers $\lceil\sqrt{\mu'}\rceil$ points—then we output the witness set as an element of the collection and delete the points of this witness set from all $O(n^2)$ ball hierarchies, in total time $2^{O(\log \mu')}n^2$ per point.

When a point is deleted from the hierarchy, it is possible that a previously considered point not placed in the top three levels can now be entered there. Therefore, after the deletion of a point we reconsider all (nondeleted) points and attempt to insert them into $O(n^2)$ ball hierarchies, in total time $2^{O(\log \mu')}n^3$ per deleted point. This implies a final runtime of $2^{O(\log \mu')}n^4$. $\square$

3.2. Bicriteria algorithm.

Given Lemma 3.4, we prove the following theorem, which is a bicriteria algorithm for the problem of removing points to achieve a target density constant. A corollary of this theorem gives a bicriteria algorithm for the problem of removing points to achieve a target doubling constant.

THEOREM 3.5. *Given a point set S with density constant $\mu(S)$ and a target density constant μ^*, let $k^* = k^*(S,\mu^*)$ be the minimum number of points that must be removed from S to obtain a set $S^* \subset S$ with density constant μ^*. Then there exists a $2^{O(\log \mu^*)}n^4$ time algorithm that removes $k' \leq \frac{c\mu^*+1}{(c-1)\mu^*+1} \cdot k^*$ points from S (for any desired $c \geq 1$) and yields a point set S' with density constant $\mu(S') \leq (c\mu^*)^2$.*

Proof. We first prove the theorem for $c = 1$. We run the algorithm of Lemma 3.4 to find a collection of distinct witness sets of size exactly $\mu^* + 1$. Remove these sets from S. It follows that the resulting set S' has density constant at most $(\mu^*)^2$. Now, in the optimal solution S^*, at least one of the points in each witness set must be removed. Hence, the algorithm removes $k' \leq (\mu^* + 1)k^*$ points.

Turning to $c > 1$, we run the algorithm of Lemma 3.4 to find a collection of distinct witness sets of size exactly $c\mu^* + 1$. Remove these sets from S. It follows that the resulting set S' has density constant at most $(c\mu^*)^2$. Now, if our algorithm has removed a witness set of size m, then in the optimal solution at least $m - \mu^*$ of these points must have been removed. It follows that the algorithm removes $k' \leq \frac{c\mu^*+1}{c\mu^*+1-\mu^*}k^* = \frac{c\mu^*+1}{(c-1)\mu^*+1} \cdot k^*$ points from S. □

This algorithm for the density constant implies a similar one for the doubling constant.

Corollary 3.6. *Given a point set S with doubling constant $\lambda(S)$ and a target doubling constant λ^*, let $m^* = m^*(S, \lambda^*)$ be the minimum number of points that must be removed from S to obtain a set S^* with doubling constant λ^*. Then there exists an $2^{O(\log \lambda^*)}n^4$ time algorithm that removes $m' \leq \frac{c(\lambda^*)^2+1}{(c-1)(\lambda^*)^2+1} \cdot m^*$ points from S (for any desired $c \geq 1$) and yields a point set S' with doubling constant $\lambda(S') \leq (c(\lambda^*)^2)^2$.*

Proof. Recall that $\sqrt{\mu(S)} \leq \lambda(S) \leq \mu(S)$, and note that a set with doubling constant λ^* has density constant at most $(\lambda^*)^2$. We apply the algorithm of Theorem 3.5 with target density constant $\mu^* = (\lambda^*)^2$, and so the resulting set has doubling constant at most $(c\mu^*)^2 \leq (c(\lambda^*)^2)^2$. □

3.3. Improved runtime.

3.3. Improved runtime. While the bicriteria algorithms implied by Theorem 3.5 and Corollary 3.6 provide a powerful trade-off for the two point removal problems, the algorithmic runtime may be undesirable for some applications. Here we present bicriteria algorithms that feature near-linear runtime at the expense of slightly higher dimension.

Theorem 3.7. *Given a point set S with density constant $\mu(S)$ and a target density constant μ^*, let $k^* = k^*(S, \mu^*)$ be the minimum number of points that must be removed from S to obtain a set S^* with density constant μ^*. Then there exists*

(i) *an algorithm that runs in $2^{O(\log \mu^*)}n \log \alpha$ time that removes $k' \leq \frac{c\mu^*+1}{(c-1)\mu^*+1} \cdot k^*$ points from S (for any desired $c \geq 1$) and yields a point set S' with density constant $\mu(S') \leq (c\mu^*)^5$;*

(ii) *an algorithm that runs in $2^{O(\log \mu^*)}n \log^3 n$ time that removes $k' \leq \frac{c\mu^*+1}{(c-1)\mu^*+1} \cdot k^*$ points from S (for any desired $c \geq 1$) and yields a point set S' with density constant $\mu(S') \leq (c\mu^*)^6$.*

Proof. We begin by building the hierarchy of [33] for S, inserting one point at a time. Now, if a point insertion causes a point of level H_{2^i} to possess more than $(c\mu^*)^5$ neighbors in H_{2^i} within distance $32 \cdot 2^i$, then we can find a witness set: By building a hierarchy for just the neighbor set on distances $\{32 \cdot 2^i, 16 \cdot 2^i, 8 \cdot 2^i, 4 \cdot 2^i, 2 \cdot 2^i, 2^i\}$, we locate in the neighbor set some witness set of size at least $c\mu^*$. (That is, some point in the neighbor set hierarchy must cover $c\mu^* + 1$ points one level down, and these points form a witness set for S.) As before, the points of the witness set are then deleted from the hierarchy of S. The algorithm terminates with set S' when no more witness sets can be found. This can all be done in $2^{O(\log \mu^*)}n \log \alpha$ time. The analysis for near-optimality of removed points is the same as above.

It is only left to show that the resulting set cannot have density constant greater than $(c\mu^*)^5$: Suppose in contradiction that S' contained a witness set of size greater than $(c\mu^*)^5$ with diameter r and minimum interpoint distance $\frac{r}{2}$. Now, each point of the witness set appears in H_1, the bottom level of the full hierarchy, and a geometric series argument gives that the distance between each point and its ancestor in level

H_{2^i} is less than 2^{i+1}. Hence, the distance between the level H_{2^i} ancestors of two different points of the witness set is greater than $\frac{r}{2} - 2^{i+2}$ and less than $r + 2^{i+2}$. Now let j be the index for which $r > 2^j \geq \frac{r}{2}$. Let $i = j - 3$, so that the distance between the ancestors of two points is greater than $\frac{r}{2} - 2^{j-1} > 0$ (and so at least 2^i) and less than $r + 2^{j-1} \leq \frac{5}{2} \cdot 2^j = 20 \cdot 2^i$. This contradicts the assumption that there does not exist in the hierarchy a set of more than $(c\mu)^5$ points within radius $32 \cdot 2^i$ and minimum interpoint distance 2^i.

The runtime of $2^{O(\log \mu^*)} n \log^3 n$ can be achieved by using the hierarchy of [24] instead of the hierarchy of [33]. (Note, however, that the semidynamic hierarchy of [15] or the static hierarchy of [27] are not sufficient for our purposes.) The analysis is similar. ◻

The above point removal algorithm for the density constant implies a similar one for the doubling constant.

COROLLARY 3.8. *Given a point set S with doubling constant $\lambda(S)$ and a target doubling constant λ^*, let $m^* = m^*(S, \lambda^*)$ be the minimum number of points that must be removed from S to obtain a set S^* with doubling constant λ^*. Then there exists*

(i) *an algorithm that runs in $2^{O(\log \lambda^*)} n \log \alpha$ time that removes $m' \leq \frac{c(\lambda^*)^2+1}{(c-1)(\lambda^*)^2+1} \cdot m^*$ points from S (for any desired $c \geq 1$) and yields a point set S' with doubling constant $\lambda(S') \leq (c(\lambda^*)^2)^5$;*

(ii) *an algorithm that runs in $2^{O(\log \lambda^*)} n \log^3 n$ time that removes $m' \leq \frac{c(\lambda^*)^2+1}{(c-1)(\lambda^*)^2+1} \cdot m^*$ points from S (for any desired $c \geq 1$) and yields a point set S' with doubling constant $\lambda(S') \leq (c(\lambda^*)^2)^6$.*

4. Applications.

The algorithms of section 3 are given a point set S and remove from S a set R, resulting in a set $S' = S - R$ of low doubling constant $(\lambda^*)^{O(1)}$. If $|R| = O(n^{1/2})$ (that is, S is nearly doubling), we can use techniques from [25, 7] to construct near-linear runtime algorithms for spanners and fast distance oracles.

We first review the spanner of [25] in section 4.1 and then present the near-linear algorithms in section 4.2

4.1. Spanner review.

We review the $(1 + \varepsilon)$-stretch spanner presented in [25] (which itself draws on the work of [19]). This spanner is constructed as follows. Given a point set S, the point hierarchy of [15] is constructed for S. Recall that each point $p \in H_{5^i}$ (for all i) is assigned as the child of a single point $q \in H_{5^{i+1}}$ that covers p (that is, the distance from p to q is at most 5^{i+1}). We add to the spanner edges connecting each parent-child pairs; these are the *parent-child edges*. Next, we add to the spanner edges to connect all point pairs $p, q \in H_{5^i}$ (for all i) if p and q are c-neighbors, that is, if $d(p, q) \leq c5^i$ for $c = \frac{25}{\varepsilon} + \frac{45}{2}$. These are the *lateral edges*. Notice that the lateral edges of level H_{5^i} are much longer than the parent-child edges of that level (by a factor of $\Theta(1/\varepsilon)$). The entire construction can be done in time $2^{O(\log \lambda(S))} n \log n + \varepsilon^{-O(\log \lambda(S))}$.

It was shown in [25] that given two points $p, q \in S$, there exists a simple spanner path that connects p, q and has stretch at most $(1 + \varepsilon)$. Let $p', q' \in H_{5^j}$ be the lowest ancestral c-neighbors of $p, q \in H_1$. (That is, j is the smallest index for which p' and q', the respective ancestors of p and q in H_{5^j}, are c-neighbors.) The low stretch spanner path is the path that begins at $p \in H_1$, follows a series of parent-child edges up to $p' \in H_{5^j}$, a single lateral edge to $q' H_{5^j}$, and a series of parent-child edges down to $q \in H_1$. The length of this path is dominated by the length of the single lateral edge: The length of the lateral edge is $\Theta(5^j/\varepsilon)$, while the length of all other edges in the path are bounded by two geometric series that each sum to $O(5^j)$. This implies a $(1 + \varepsilon)$-stretch spanner path for the pair p, q.

More precisely, $d(p,p') \geq \sum_{k=1}^{j} 5^k < \frac{5}{4}5^j$ (and similarly for $d(q,q')$), and so the spanner path distance from p,q is less than $d(p',q') + \frac{5}{2}5^j$. At the same time, $d(p,q) \geq d(p',q') - d(p,p') - d(q,q') > d(p',q') - \frac{5}{2}5^j$, implying a spanner stretch less than $\frac{d(p',q')+\frac{5}{2}5^j}{d(p',q')-\frac{5}{2}5^j} = 1 + \frac{5\cdot 5^j}{d(p',q')-\frac{5}{2}5^j}$. The proof is completed by demonstrating that $d(p',q') > \frac{5\cdot 5^j}{\varepsilon} + \frac{5}{2}5^j$: Consider points $p'', q'' \in H_{5^{j-1}}$, the respective children of $p', q' \in H_{5^j}$ that are ancestors of $p, q \in H_1$. Since p'', q'' are not connected by a lateral edge, we have that $d(p'',q'') > c5^{j-1}$, and so $d(p',q') \geq d(p'',q'') - d(p',p'') - d(q',q'') > c5^{j-1} - 2\cdot 5^j = \frac{5\cdot 5^j}{\varepsilon} + \frac{5}{2}5^j$.

4.2. Near-linear algorithms. In this section we present near-linear algorithms for nearly-doubling spaces. We have the following theorem.

THEOREM 4.1. *There exists an algorithm that, given point sets S' and R ($|R| = O(\sqrt{|S'|})$), builds a $(1+\varepsilon)$-stretch spanner for $S = S' \cup R$ ($n = |S|$) with $\varepsilon^{-O(\log \lambda^*)}n$ edges in $2^{O(\log \lambda^*)}n \log n + \varepsilon^{-O(\log \lambda^*)}n$ time.*

Proof. We first construct the full graph for R, which is a 1-stretch spanner of $O(n)$ edges for these points, in time $O(R^2) = O(n)$. We then construct a $(1+\varepsilon)$ spanner for S' in the manner described above, in time $2^{O(\log \lambda^*)}n \log n + \varepsilon^{-O(\log \lambda^*)}n$. It is left only to guarantee $(1+\varepsilon)$ stretch between the points of S' and R. To this end, for each point $p \in R$ we locate the lowest hierarchical level H_{2^i} of S' in which p is covered and connect p to its covering point with a parent-child edge and to all points of levels H_{2^i} and below within distance $c2^i$ using lateral edges. As in [25], there are $\varepsilon^{-O(\log \lambda)}$ edges incident on p, and this construction mimics an actual insertion of p into the hierarchy, which can be done in time $2^{O(\log \lambda^*)} \log n + \varepsilon^{-O(\log \lambda^*)}$ per insertion. It follows that there exists low stretch paths connecting $p \in R$ to all points of S'. □

The following corollary is a consequence of the spanner construction of Theorem 4.1.

COROLLARY 4.2. *There are algorithms that, given point sets S' and R ($|R| = O(\sqrt{|S'|})$), compute*

(i) *a $(1+\varepsilon)$-approximation to the MST for S, in time $2^{O(\log \lambda^*)}n \log n + \varepsilon^{-O(\log \lambda^*)}n$,*

(ii) *a $(1+\varepsilon)$-approximation to all points nearest neighbor in time $2^{O(\log \lambda^*)}n \log n + \varepsilon^{-O(\log \lambda^*)}n$,*

(iii) *a $(1+\varepsilon)$-approximate distance oracle that supports distance queries in $O(1)$ expected time with storage $2^{O(\log \lambda^* \log\log \lambda^*)}n + \varepsilon^{-O(\log \lambda^*)}n$ and construction time $2^{O(\log \lambda^*)}n \log n + 2^{O(\log \lambda^* \log\log \lambda^*)}n + \varepsilon^{-O(\log \lambda^*)}n$.*

Proof. (i) Given the above $(1+\varepsilon)$-stretch spanner, an MST for the spanner can be constructed by running Prim's algorithm while storing the edges in a Fibonacci heap. For a spanner of n points and $\varepsilon^{-O(\log \lambda^*)}n$ edges, this can all be done in time $O(n \log n) + \varepsilon^{-O(\log \lambda^*)}n$ [16], so the total runtime is dominated by the spanner construction.

The MST for the spanner is a $1+\varepsilon$ approximation to the MST for S. Let E be the edge set constructed as follows. For each edge e connecting points p,q in the MST of S, add to E all edges of the minimum weight spanner path connecting p,q (not already added to E). Clearly, E spans all points of S, while the sum of edge weights in E is at most $1+\varepsilon$ times the weight of the MST. The MST of the spanner must possess this weight guarantee as well.

(ii) A $(1+\varepsilon)$-approximate nearest neighbor for each point p may be found by consulting the spanner edges incident on p and choosing the closest incident point. This can be maintained in $O(1)$ time per edge insertion.

(iii) A $(1+\varepsilon)$-approximate distance oracle with expected $O(1)$ query time for doubling spaces was presented in [7]. This oracle was built on the spanner of [25] described above. The structure records the exact distance between any pair of points that are connected in the spanner. For query points p and q, the algorithm simply locates the lowest ancestral c-neighbors p', q' of p, q, and returns their distance. (Recall that p', q' are connected by a lateral edge, so their true distance is recorded in the spanner.) It follows from the spanner analysis above that the distance between p' and q' is a $(1+\varepsilon)$-approximation to the distance between p and q.

We extend the construction of [25] to the spanner in the proof of Theorem 4.1: We record the distance between any pairs of points that are connected in the spanner of the proof of Theorem 4.1. It follows that the distance between any pair $p, q \in R$ is recorded explicitly. For all pairs $p, q \in S'$, a search for the lowest ancestral c-neighbors of p and q returns an approximation for the distance between p and q. For points $p \in R$ and $q \in S'$, their distance can similarly be derived via a search for the lowest ancestral c-neighbors: If p and q are themselves c-neighbors, then they were connected by a spanner edge (in the construction for the proof of Theorem 4.1) and their distance is stored explicitly. Otherwise, we query the oracle for the lowest c-neighbor of q and the parent of p (that is, the point in S' that covers p). $\square$

Acknowledgments. The authors thank Uri Feige and Liam Roditty for useful discussions and thank the anonymous referees for valuable suggestions.

REFERENCES

[1] I. Abraham, Y. Bartal, and O. Neiman, *Local embeddings of metric spaces*, in Proceedings of 39th Annual ACM Symposium on Theory of Computing, 2007, pp. 631–640.

[2] I. Abraham, Y. Bartal, and O. Neiman, *Embedding metric spaces in their intrinsic dimension*, in 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 2008, pp. 363–372.

[3] I. Abraham, C. Gavoille, A. V. Goldberg, and D. Malkhi, *Routing in networks with low doubling dimension*, in 26th IEEE International Conference on Distributed Computing Systems, 2006, p. 75.

[4] M. R. Ackermann, J. Blömer, and C. Sohler, *Clustering for metric and non-metric distance measures*, in Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 2008, pp. 799–808.

[5] P. Assouad, *Plongements lipschitziens dans* $\mathbf{R}^n$, Bull. Soc. Math. France, 111 (1983), pp. 429–448.

[6] Y. Bartal, L. Gottlieb, and R. Krauthgamer, *The traveling salesman problem: Low dimensionality implies a polynomial time approximation scheme*, in Proceedings of 44th Annual ACM Symposium on Theory of Computing, 2012.

[7] Y. Bartal, L.-A. Gottlieb, T. Kopelowitz, M. Lewenstein, and L. Roditty, *Fast, precise and dynamic distance queries*, in Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, 2011, pp. 840–853.

[8] Y. Bartal, B. Recht, and L. J. Schulman, *Dimensionality reduction: Beyond the Johnson-Lindenstrauss bound*, in Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, 2011, pp. 868–887.

[9] A. Beygelzimer, S. Kakade, and J. Langford, *Cover trees for nearest neighbor*, in Proceedings of the 23rd International Conference on Machine Learning, 2006, pp. 97–104.

[10] N. H. Bshouty, Yi Li, and P. M. Long, *Using the doubling dimension to analyze the generalization of learning algorithms*, J. Comput. System Sci., 75 (2009), pp. 323–335.

[11] H. Chan, A. Gupta, and K. Talwar, *Ultra-low-dimensional embeddings for doubling metrics*, in Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 2008, pp. 333–342.

[12] T.-H. Chan and A. Gupta, *Small hop-diameter sparse spanners for doubling metrics*, in Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, 2006, pp. 70–78.

[13] T.-H. Chan and A. Gupta, *Approximating TSP on metrics with bounded global growth*, in Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 2008, pp. 690–699.

[14] M. Chlebík and J. Chlebíková, *Approximation hardness of dominating set problems in bounded degree graphs*, Inform. and Comput., 206 (2008), pp. 1264–1275.

[15] R. Cole and L.-A. Gottlieb, *Searching dynamic point sets in spaces with bounded doubling dimension*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing, 2006, pp. 574–583.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms,* 3rd ed., MIT Press, Cambridge, MA, 2009.

[17] M. Damian, S. Pandit, and S. V. Pemmaraju, *Distributed spanner construction in doubling metric spaces*, in Proceedings of OPODIS, 2006, pp. 157–171.

[18] S. A. Friedler and D. M. Mount, *Approximation algorithm for the kinetic robust k-center problem*, Comput. Geom. Theory Appl., 43 (2010), pp. 572–586.

[19] J. Gao, L. J. Guibas, and A. Nguyen, *Deformable spanners and applications*, Comput. Geom. Theory Appl., 35 (2006), pp. 2–19.

[20] A. Gionis, A. Hinneburg, S. Papadimitriou, and P. Tsaparas, *Dimension induced clustering*, in KDD '05: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 2005, pp. 51–60.

[21] L.-A. Gottlieb, A. Kontorovich, and R. Krauthgamer, *Efficient classification for metric data*, in Proceedings of COLT, 2010.

[22] L.-A. Gottlieb and R. Krauthgamer, *Proximity algorithms for nearly-doubling spaces*, in 13th International Workshop on Approximation, Randomization, and Combinatorial Optimization, Lecture Notes in Comput. Sci. 6302, Springer, New York, 2010, pp. 192–204.

[23] L.-A. Gottlieb and R. Krauthgamer, *A nonlinear approach to dimension reduction*, in Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, 2011, pp. 888–899.

[24] L.-A. Gottlieb and L. Roditty, *Improved algorithms for fully dynamic geometric spanners and geometric routing*, in Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 2008, pp. 591–600.

[25] L.-A. Gottlieb and L. Roditty, *An optimal dynamic spanner for doubling metric spaces*, in ESA '08: Proceedings of the 16th Annual European Symposium on Algorithms, 2008, pp. 478–489.

[26] A. Gupta, R. Krauthgamer, and J. R. Lee, *Bounded geometries, fractals, and low-distortion embeddings*, in Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, 2003, pp. 534–543.

[27] S. Har-Peled and M. Mendel, *Fast construction of nets in low dimensional metrics, and their applications*, in Proceedings of the 21st Annual Symposium on Computational Geometry, ACM, 2005, pp. 150–158.

[28] S. Har-Peled and M. Mendel, *Fast construction of nets in low-dimensional metrics and their applications*, SIAM J. Comput., 35 (2006), pp. 1148–1184.

[29] J. Hastad, *Clique is hard to approximate within $n^{1-\varepsilon}$*, in Acta Math., 1996, pp. 627–636.

[30] J. M. Kleinberg, A. Slivkins, and T. Wexler, *Triangulation and embedding using small sets of beacons*, in Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, 2004, pp. 444–453.

[31] G. Konjevod, A. W. Richa, and D. Xia, *Dynamic routing and location services in metrics of low doubling dimension*, in DISC '08: Proceedings of the 22nd International Symposium on Distributed Computing, 2008, pp. 379–393.

[32] G. Konjevod, A. W. Richa, D. Xia, and H. Yu, *Compact routing with slack in low doubling dimension*, in Proceedings of the 26th Annual ACM Symposium on Principles of Distributed Computing, ACM, 2007, pp. 71–80.

[33] R. Krauthgamer and J. R. Lee, *Navigating nets: Simple algorithms for proximity search*, in Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, Jan. 2004, pp. 791–801.

[34] C. H. Papadimitriou and M. Yannakakis, *Optimization, approximation, and complexity classes*, J. Comput. System Sci., 43 (1991), pp. 425–440.

[35] M. Parnas and D. Ron, *Testing metric properties*, Inform. and Comput., 187 (2003), pp. 155–195.

[36] R. Raz and S. Safra, *A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP*, in Proceedings of the 29th Annual ACM Symposium on Theory of Computing, ACM, 1997, pp. 475–484.

[37] A. Slivkins, *Distance estimation and object location via rings of neighbors*, in Proceedings of the 24th Annual ACM Symposium on Principles of Distributed Computing, 2005, pp. 41–50.

[38] K. Talwar, *Bypassing the embedding: Algorithms for low dimensional metrics*, in Proceedings of the 36th Annual ACM Symposium on Theory of Computing, 2004, pp. 281–290.