The black-box complexity of nearest neighbor search

Robert Krauthgamer[1] * and James R. Lee[2] **

[1] IBM Almaden
[2] U. C. Berkeley

Abstract. We define a natural notion of efficiency for approximate nearest-neighbor (ANN) search in general n-point metric spaces, namely the existence of a randomized algorithm which answers $(1+\varepsilon)$-approximate nearest neighbor queries in polylog(n) time using only polynomial space. We then study which families of metric spaces admit efficient ANN schemes in the black-box model, where only oracle access to the distance function is given, and any query consistent with the triangle inequality may be asked.

For $\varepsilon < \frac{2}{5}$, we offer a complete answer to this problem. Using the notion of metric dimension defined in [GKL03] (à la [Ass83]), we show that a metric space X admits an efficient $(1+\varepsilon)$-ANN scheme for any $\varepsilon < \frac{2}{5}$ *if and only if* $\dim(X) = O(\log\log n)$. For coarser approximations, clearly the upper bound continues to hold, but there is a threshold at which our lower bound breaks down—this is precisely when points in the "ambient space" may begin to affect the complexity of "hard" subspaces $S \subseteq X$. Indeed, we give examples which show that $\dim(X)$ does not characterize the black-box complexity of ANN above the threshold.

Our scheme for ANN in low-dimensional metric spaces is the first to yield efficient algorithms without relying on any additional assumptions on the input. In previous approaches (e.g., [Cla99,KR02,KL04]), even spaces with $\dim(X) = O(1)$ sometimes required $\Omega(n)$ query times.

1 Introduction

Nearest-neighbor search. Nearest-neighbor search (NNS) is the problem of preprocessing a set X of n points lying in a huge (possibly infinite) metric space (M, d) so that given a query $q \in M$, one can efficiently locate the nearest point to q among the points in X. Computing such nearest neighbors efficiently is a classical and fundamental problem with numerous practical applications. These

* IBM Almaden Research Center, 650 Harry Road, San Jose CA 95120, USA. Part of this work was done while this author was with the International Computer Science Institute and with the Computer Science Division of U.C. Berkeley. Email: robi@almaden.ibm.com

** Computer Science Division, U.C. Berkeley, Berkeley, CA 94720, USA. Supported by NSF grant CCR-0121555 and an NSF Graduate Research Fellowship. Email: jrl@cs.berkeley.edu

include data compression, database queries, machine learning, computational biology, data mining, pattern recognition, and ad-hoc networks. A common feature of many of these examples is that comparing two elements is costly, hence the number of distance computations should be made as small as possible.

Most previous research has focused on the important special case when $M = \mathbb{R}^d$ and distances are computed according to some ℓ_p norm. While many types of data can be naturally represented in such a form, this is certainly not true for a significant number of applications, and it is therefore desirable to address NNS in general metric spaces. On the other hand, data structures for general metrics might perform a nearest neighbor query in time as poorly as $\Omega(n)$ which is unacceptable in practice. Such a dependence is inherent even when only approximate solutions are required. A well-known example is where X forms a uniform metric, so that the interpoint distances in X are all equal, providing essentially no information.

Metric dimension. Given this state of affairs, an increasing amount of recent attention has focused on understanding the complexity of NNS in terms of a metric's implicit structure. In Euclidean spaces, an obvious and common measure for a metric's complexity is the dimension of the Euclidean host space. Thus it is natural that to characterize the complexity of general metric spaces, one ought to define an analogous notion of *metric dimension,* and indeed this approach has been pursued to great success in recent papers [Cla99,KR02,KL04,HKMR04], where significant progress on solving exact and approximate versions of the NNS problem in general metrics has been made.

Unfortunately, each of these works falls short of offering the sort of generality that one should desire from such an approach. In [Cla99], to achieve efficient algorithms (for *exact* NNS), it is necessary to make strong assumptions about the distribution of queries. In [KR02,HKMR04], the notion of dimension is too restrictive, eliminating large classes of metric spaces which should be considered low-dimensional, and for which efficient algorithms should exist (see [KL04] for a more detailed explanation).

Finally, in [KL04], a more satisfying notion of dimension (taken from [GKL03], and independently used in a different form by [Cla99]) is proposed, but the algorithms in both [KL04] and [Cla99] are efficient only under the additional assumption that the *aspect ratio* Φ (i.e. the ratio of the largest to smallest distance in X) is at most polynomial in $n = |X|$. In particular, the algorithm presented in [KL04] answers approximate nearest neighbor queries in time $2^{O(\dim(X))} \log \Phi$. Thus even when the set of points is $X = \{1, 2, 4, \ldots, 2^n\} \subseteq \mathbb{R}$ with the line metric $d(x, y) = |x - y|$, as in Figure 1(a), the algorithms of [KL04], as well as those of [Cla99,KR02,HKMR04], require $\Omega(n)$ time to answer some queries (i.e. they are no better than the trivial algorithm which tests every point). Despite the fact that (X, d) is clearly "low-dimensional" (being a subset of the real line), previous approaches perform dismally. Besides being theoretically disappointing, these algorithms are incapable of searching for (even approximate) nearest neighbors in highly clustered data (e.g. 1(b)).

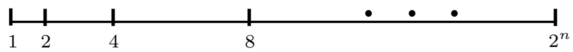
Efficient algorithms in the black-box model. In the present work, we are concerned with *approximate nearest neighbor search* (ANN). The $(1+\varepsilon)$-ANN problem is defined as follows: Given a query $q \in M$, we are required to return an element $a \in X$ for which $d(q,a) \leq (1+\varepsilon)\, d(q,X)$, where $d(q,X)$ is the distance from q to the closest point in X. (This is after an initial preprocessing stage.) We resolve the aforementioned shortcomings by presenting an ANN data structure for general metric spaces which is efficient whenever $\dim(X)$ (defined formally in Section 1.2) is small, and under no additional assumptions.

We will discuss our definition of "efficient" momentarily. Beforehand, let us describe another question that arises in the above framework: Is "dimension" the right notion to characterize the complexity of ANN in general metric spaces? Certainly one may motivate the study of algorithms for "low-dimensional" metrics by their abundance in practical settings (see [KL04]), but one should also consider how tightly $\dim(X)$ captures the difficulty of nearest neighbor search in general metrics. To this end, we consider a black-box model of nearest neighbor search in arbitrary metric spaces, where the query is accessed as an oracle via the distance function. We say that a metric space X admits an efficient $(1+\varepsilon)$-ANN scheme if there exists a (possibly randomized) algorithm which answers any possible $(1+\varepsilon)$-approximate nearest neighbor query in $\mathrm{polylog}(n)$ time using only polynomial space (although quite a natural definition, we offer additional motivation in Section 1.2, where the model is specified more precisely).

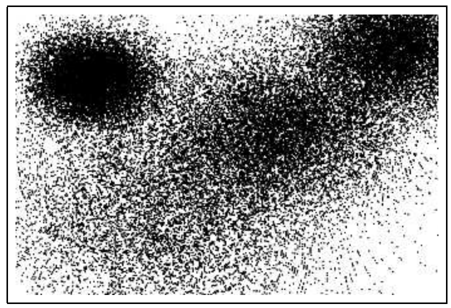
Under this complexity regime, we show that for every $\varepsilon < \frac{2}{5}$, a metric space X admits an efficient $(1+\varepsilon)$-ANN scheme *if and only if* $\dim(X) = O(\log\log n)$, and thus below a certain approximation threshold, $\dim(X)$ captures precisely the complexity of the problem. The behavior above the threshold is quite different, and we demonstrate in Section 2 that for sufficiently coarse approximations, the "ambient space" begins to play a crucial role, at which point even metrics of very high dimension may become tractable. We note that the constants of these constructions are not optimized; our primary goal is simply to show the existence of an approximation threshold, on either side of which drastically different behaviors are exhibited.

1.1 Results and techniques

Upper bounds. We give the first efficient (i.e. $\mathrm{polylog}(n)$ query time and $\mathrm{poly}(n)$ space) $(1+\varepsilon)$-ANN scheme for metrics whose doubling dimension, denoted by $\dim(X)$ (and defined in Section 1.2), is small. In particular, these bounds hold for any fixed $\varepsilon > 0$ and whenever $\dim(X) = O(\log\log n)$. For instance, in the case where $\dim(X) = O(1)$, our algorithm answers queries in $O(\log^2 n)$ time and $O(n^2)$ space, while making only $O(\log n)$ calls to the distance function. We note that the space bound we achieve for arbitrary metrics—obtained in Section 3.2—is a polynomial whose degree is *independent* of $\dim(X)$ and the desired approximation. Indeed, our data structure can be built without knowledge of ε (which can be passed as a parameter at query-time). When $\dim(X)$ is small, a general reduction from $O(1)$-ANN to $(1+\varepsilon)$-ANN follows easily from the techniques of [KL04], which we review in Section 1.2.



(a) The doubling line



(b) Mixture of gaussians in the plane

Fig. 1. Clustered point sets

Our data structure is based on two new techniques. The first is a structural theorem about the existence of "dense," "well-separated" clusters of points in low-dimensional metrics. These sets manifest themselves in the form of *ring separators*— "thick" annuli whose inside and outside each contain a large fraction of the points. (A similar object is used in the construction of the ring-cover trees of [IM98] which are used to solve ANN in $\mathbb{R}^d$. Our setting is quite different, hence the corresponding structural theorems are distinct.) Using these separators, we build a binary decision tree of height $2^{O(\dim(X))} \log n$ which can be used to answer $O(1)$-ANN queries in time $2^{O(\dim(X))} \log n$. Unfortunately, the natural implementation of this tree requires space $n^{2^{\dim(X)}}$, which is $n^{\Omega(\text{polylog}(n))}$ even when $\dim(X) = \Theta(\log \log n)$.

This exponential blow-up in space is a typical problem encountered in NNS algorithms based on metric decompositions, and is the most difficult technical challenge faced by the present work. In Section 3.2, we overcome this problem for low-dimensional metrics, obtaining a data structure that uses $O(n^2 \,\text{polylog}(n))$ space whenever $\dim(X) = O(\log \log n)$. In addition, even for arbitrary spaces (with no bound on the dimension), the space consumption is only polynomial. This improvement requires a second new technique which amounts to "folding" the decision tree back onto itself, often merging many apparently distinct branches into one. The difficulties and solutions are discussed more thoroughly in Section 3.2. This folding allows us to obtain a very compact "approximate" representation of the previously huge decision tree, while incurring only a small additional overhead at every node.

We note that since the doubling dimension was introduced in [GKL03], and the premise that "low-dimensional" general metrics should be algorithmically tractable was set forth, an increasing number of works have found applications of this idea to optimization problems; we mention, in particular, the predecessor to this work [KL04] and the new results of [Tal04] for approximating problems like TSP in low-dimensional metrics. We also mention the related work of [KKL03] in machine learning. We believe that the development and discovery of the rich properties of low-dimensional metrics continued herein will find additional application elsewhere.

Lower bounds. Our lower bounds are entirely information theoretic. Given a metric space X, there exists a "nearly-uniform" submetric $S \subseteq X$ whose size is roughly $k \geq 2^{\Omega(\dim(X))}$. Nearly uniform means that the aspect ratio (the ratio of the largest pairwise distance in S to the smallest) is bounded by a small constant. In Section 2, we then prove that, for every $\delta > 0$, this "large" subset S must contain within itself a subset $T \subseteq S$ with very small aspect ratio, i.e. $2 + \delta$, and yet which satisfies $|T| \geq k^{\delta'}$ (where δ' depends on δ, of course). This is a very simple (yet interesting) Ramsey-like property of metric spaces.

Now, if an algorithm were not allowed to compute distances from the query to $X \setminus T$ (i.e. the "ambient space"), then a lower bound of $\Omega(k^{\delta'})$ queries for $(1+\delta)$-ANN would follow fairly easily for T. And indeed, by a slightly technical extension argument, we can prove that any algorithm solving the $(1+\varepsilon)$-ANN problem must make at last $2^{\Omega(\dim(X))}$ queries to the distance oracle for $\varepsilon < \frac{2}{5}$. This shows that in the black-box model, querying against the ambient space cannot help too much when one requires a sufficiently fine approximation.

But our lower bound breaks down for coarser approximations, and we show that this is for good reason: When only a 3-approximation is desired, there are n-point metrics X with $\dim(X) = \Omega(\log n)$ for which every query against X can be decided in $O(\log n)$ time in the black-box model. Thus above a certain approximation threshold, $\dim(X)$ no longer characterizes the complexity of ANN.

1.2 Preliminaries

Metric spaces. Let (X, d) be an n-point metric space, and let $S \subseteq X$ be a subset. We denote by

$$B_S(x, r) = \{y \in S : d(x, y) < r\}$$

the open ball of radius r about x in S. When $S = X$, we omit the subscript S. We write $d(x, S) = \inf_{y \in S} d(x, y)$. Define $\mathrm{diam}(S) = \sup_{x,y \in S} d(x, y)$, and let the *aspect ratio* of S be the quantity

$$\Phi(S) = \frac{\mathrm{diam}(S)}{\inf_{x,y \in S} d(x, y)}.$$

Finally, we say that a subset Y of X is a *β-net* if it satisfies (1) For every $x, y \in Y, d(x, y) \geq \beta$ and (2) $X \subseteq \bigcup_{y \in Y} B(y, \beta)$. Such nets always exist for any

$\beta > 0$. For finite metrics, they can be constructed greedily. For arbitrary metrics, proof of their existence is an easy application of Zorn's lemma.

The doubling dimension. We recall that the *doubling constant* $\lambda(X)$ is the least value λ such that every ball in X can be covered by λ balls of half the radius. The *doubling dimension* [GKL03] is then defined by $\dim(X) = \log_2 \lambda(X)$. Here are some simple properties which demonstrate that $\dim(X)$ is a robust and meaningful notion.

1. For $X = \mathbb{R}^k$ equipped with any norm, $\dim(X) = \Theta(k)$.
2. If $S \subseteq X$, then $\dim(S) \leq 2 \cdot \dim(X)$.
   (Using a slightly different definition of $\dim(X)$ which is equivalent up to a constant, one can ensure that $\dim(S) \leq \dim(X)$.)
3. $\dim(X_1 \cup \cdots \cup X_m) \leq \max_i \dim(X_i) + \log m$.
   (In particular, $\dim(X) \leq \log |X|$.)

The following simple lemma is important.

Lemma 1 (Nearly-uniform metrics). *Let (X, d) be a metric space, and let $S \subseteq X$. If the aspect ratio of the metric induced on S is at most $\Phi \geq 2$, then $|S| \leq \Phi^{O(\dim(X))}$.*

Proof. Let $d_{\min} = \inf\{d(x, y) : x, y \in S\}$ and $d_{\max} = \sup\{d(x, y) : x, y \in S\}$ be the minimum and maximum interpoint distance in S, respectively, and assume that $\Phi = \frac{d_{\max}}{d_{\min}} < \infty$. Notice that S is contained in a ball of radius $2\, d_{\max} \leq 2\Phi\, d_{\min}$ in X (centered at any point of S). Applying the definition of doubling dimension iteratively several times we get that this ball, and in particular S, can be covered by $2^{\dim(X) \cdot O(\log \Phi)}$ balls of radius $d_{\min}/3$. Each of these balls can cover at most one point of S (by definition of $d_{\min}$) and thus $|S| \leq 2^{\dim(X) \cdot O(\log \Phi)} \leq \Phi^{O(\dim(X))}$.

In particular, we observe that the above lemma provides a bound on the cardinality of a δR-net intersected with a ball of radius R. Namely, such an intersection contains at most $(\frac{1}{\delta})^{O(\dim(X))}$ points.

The black-box model and efficiency. Our model is quite simple. Suppose that (X, d) is a metric space. We assume that the only thing known about the query (and thus the only constraint on the query) is that the space $(X \cup \{q\}, d)$ is again a metric space, i.e. that the query does not violate the triangle inequality. The only access that an algorithm has to the query is through oracle calls to the distance function, i.e. the values $d(q, x)$ for $x \in X$. We assume that $d(q, \cdot)$ can be evaluated in unit time (although this is without loss of generality, since our upper bounds scale linearly with the time needed to evaluate the distance function, and our lower bounds are in terms of the number of calls to $d(q, \cdot)$).

We are defining an algorithm as "efficient" if, after the preprocessing phase, it can answer any query in $\mathrm{polylog}(n)$ time using only $\mathrm{poly}(n)$ space. We don't make any restriction on preprocessing time or space, but we note that in all of our upper bounds, both are linear in the space used by the algorithm for answering a query.

As for the running time, we note that all of the algorithms in [Cla99,KR02,KL04] strive for polylog(n) query times, thus it is *the* natural candidate for "efficiency." We also note that the best algorithms for ANN in high-dimensional Euclidean spaces answer queries in polylog(n) time [IM98,KOR98,H01]. As for space, poly(n) is again the natural choice, but this assumption should not be abused. Straightforward implementations of the algorithms of [IM98] and [KOR98], although outstanding theoretical achievements, are hampered due to their extremely high space complexity (the degree of the polynomial grows with $\frac{1}{\varepsilon}$ for $(1+\varepsilon)$-ANN).

Even in the worst case (i.e. when $\dim(X) = \Omega(\log n)$), the algorithms of Section 3.2 use only poly(n) space (independent of the approximation factor desired). When $\dim(X) = O(\log\log n)$, the space consumption is $O(n^2 \operatorname{polylog}(n))$. This factor has not been optimized, and we hope that eventually a near-linear space algorithm can be obtained, at least for the case when $\dim(X) = O(1)$.

The [KL04] reduction to $O(1)$-ANN. In [KL04], it is shown that, using only $2^{O(\dim(X))} \cdot n$ space, one can maintain a data structure which, given a query q, converts any $(1+\alpha)$-ANN to q to a $(1+\varepsilon)$-ANN using time $(\alpha/\varepsilon)^{O(\dim(X))}$. (In essence, one can do "brute-force" search around the $O(1)$-ANN. The number of nearby points in a $(1+\varepsilon)$-net is only exponential in the doubling dimension.) This term is only polylog(n) whenever $\dim(X) = O(\log\log n)$ and $\alpha/\varepsilon = O(1)$, thus we content ourselves with finding $O(1)$-ANNs in everything that follows. One actually needs to maintain pointers from the data structure in the current paper, to that borrowed from [KL04], but this is a minor issue which we ignore in the current version.

2 Lower bounds

In this section, we show that for any metric space X and any fixed $\varepsilon < \frac{2}{5}$, solving the $(1+\varepsilon)$-ANN problem on X is as hard as unordered search in a k-element database with $k = 2^{\Omega(\dim(X))}$. It will follow that any algorithm (deterministic or randomized) which solves the $(1+\varepsilon)$-ANN problem on X must make at least $2^{\Omega(\dim(X))}$ calls to the distance oracle for some query q. We note that the constructions of this section are not optimized; our goal is simply to show the existence of an approximation threshold, on either side of which drastically different behaviors are exhibited.

Theorem 1. *For any metric space X and any fixed $\varepsilon < \frac{2}{5}$, any algorithm solving the $(1+\varepsilon)$-ANN problem on X must make at least $2^{\Omega(\dim(X))}$ calls to the distance oracle for some query q. For randomized algorithms, this bound holds in expectation.*

First, we require a partial converse to Lemma 1.

Lemma 2. *For any n-point metric space X and any $2 < \Phi_0 \leq 4$, there exists a subset $S \subseteq X$ with $\Phi(S) \leq \Phi_0$ and $|S| \geq 2^{\dim(X)(\log \Phi_0 - 1)}$.*

The full proof is deferred to the full version. The basic idea is that, if every subset of small aspect ratio is small, this yields a recipe for covering every large ball by smaller balls (and hence the dimension must be small).

Theorem 2. *Let (X, d) be any metric space which contains a submetric $S \subseteq X$ with $\Phi(S) \leq \Phi_0$ and $|S| \geq k$. Then for every $\varepsilon < (\Phi_0 + \frac{1}{2})^{-1}$, any algorithm for $(1+\varepsilon)$-ANN on X must make at least $\Omega(k)$ calls to the distance oracle for some query q. For randomized algorithms, this holds in expectation.*

Proof (Sketch). Let $S = \{x_1, x_2, \ldots, x_k\}$. Let $d_{\max} = \max_{x,y \in S} d(x, y)$ and $d_{\min} = \min_{x,y \in S} d(x, y)$. To each index $i \in \{1, \ldots, k\}$, we associate a query q_i which satisfies:

$$d(q_i, y) = \begin{cases} \frac{1}{2}d_{\max} + \frac{1}{4}d_{\min} + d(y, x_i) & \text{if } d(y, x_i) < \frac{1}{2}d_{\min} \\ \frac{1}{2}d_{\max} + \frac{3}{4}d_{\min} & \text{if } d(y, x_i) \geq \frac{1}{2}d_{\min} \text{ and } d(y, S) \leq \frac{1}{2}d_{\min} \\ \frac{1}{2}d_{\max} + \frac{1}{4}d_{\min} + d(y, S) & \text{otherwise} \end{cases}$$

First, one must assure that the space $(X \cup \{q_i\}, d)$ satisfies the triangle inequality for every $1 \leq i \leq k$. Then, one shows that for $\varepsilon = \varepsilon(\Phi_0)$ small enough, finding a $(1+\varepsilon)$-ANN to q_i is equivalent to guessing the value of i. The lower bound of $\Omega(k)$ follows. These proofs are deferred to the full version.

Now we prove the main theorem of this section.

Proof (of Theorem 1). Let (X, d) be any metric space. As $\Phi_0 \to 2$ in Lemma 2, the lower bound value of ε to which the preceding theorem applies behaves like $\varepsilon < (\Phi_0 + \frac{1}{2})^{-1} \to \frac{2}{5}$. Thus for any fixed $\varepsilon < \frac{2}{5}$, there is a lower bound of $2^{\Omega(\dim(X))}$ on the number of calls to the distance function which are needed to answer some $(1+\varepsilon)$-ANN query.

We obtain the following corollary.

Corollary 1. *If $\dim(X) = \omega(\log \log n)$ and $\varepsilon < \frac{2}{5}$, then there is no efficient $(1+\varepsilon)$-ANN scheme for X in the black-box model, since $2^{\Omega(\dim(X))}$ is bigger than any* $\mathrm{polylog}(n)$.

2.1 Above the threshold

In this section, we show that when coarser approximations are desired, there are metrics of high dimension which nevertheless admit very efficient ANN algorithms, and thus the lower bounds of the previous section cannot be pushed too much further. Again, we do not seek to optimize constants.

Let $A = \{e_1, \ldots, e_n\}$ where $e_i \in \mathbb{R}^n$ is an n-dimensional vector with a 1 in the ith coordinate and zeros elsewhere. Additionally, let B consist of $k = O(\log n)$ vectors chosen at random by setting each of their coordinates to be -1 or $+1$ with equal probabilities (independent of all other coordinates). We endow these points with the ℓ_∞ metric, i.e., for any two vectors $u, v \in \mathbb{R}^n$, let $d(u, v) = \|u - v\|_\infty = \max_{1 \leq i \leq n} |u_i - v_i|$ (where v_i is the ith coordinate of v).

Let $X = A \cup B$ be the set of points to be preprocessed. Clearly, X contains the n-point uniform metric A, and thus $\dim(X) \geq \dim(A) = \log n$. On the other hand, $|X| \leq 2n$ and thus $\dim(X) \leq 1 + \log n$. However, it is not difficult to verify that, with high probability (over the choice of B), there exists a 3-ANN algorithm for X. We omit the description from this version.

3 Efficient algorithms

We provide two algorithms for $(1+\varepsilon)$ approximate nearest neighbor search in a general metric space; the two have similar query time, but they differ in their space requirement. By the general reduction discussed in Section 1.2, it suffices to exhibit an $O(1)$-ANN algorithm. Our first algorithm (Section 3.1) is based on the existence of a certain ring-separator, which naturally yields a binary decision tree that can be used to solve 3-ANN. The decision tree's depth is $2^{O(\dim(X))} \log n$, so this algorithm has an excellent (actually optimal) query time. However, it's space requirement grows rapidly with $\dim(X)$. The second algorithm, which achieves space that is polynomial in n (independently of $\dim(X)$) is significantly more complex, and we refer the reader to Section 3.2 for a discussion of the subtle issues which arise.

3.1 The ring-separator tree

The basic notion introduced in this subsection is that of a ring-separator; this naturally yields a ring-separator tree, which can be used as a binary decision tree for 3-ANN. Throughout this section, we shall use the following definition. For $x \in S \subseteq X$ and $R_1, R_2 \geq 0$, define the *annulus about* x as

$$A_S(x, R_1, R_2) \stackrel{\text{def}}{=} B_S(x, R_2) \setminus B_S(x, R_1).$$

The ring-separator. Let (X, d) be an n-point metric space. A δ*-ring-separator* of a subset $S \subseteq X$ is a pair (x, R) consisting of a point $x \in S$ and a real $R \geq 0$, that satisfies the following condition: $|B_S(x, R)| \geq \delta |S|$ and yet $|B_S(x, 2R)| \leq (1-\delta)|S|$. We now prove the main lemma of this subsection.

Lemma 3 (Ring separators). *For any metric space (X, d) and any subset $S \subseteq X$ with $|S| \geq 2$, there exists a δ-ring-separator of S with $\delta \geq (\frac{1}{2})^{O(\dim(X))}$.*

Proof. We proceed by contradiction. Fix some $0 < \delta < 1$ and assume that S does not have a δ-ring-separator; we will show that for a sufficiently large constant $c > 0$, $\delta > (\frac{1}{2})^{c \dim(X)}$, thus proving the lemma.

Let $\bar{B}_S(x, r) = \{y \in S : d(x, y) \leq r\}$ be the *closed ball* of radius r around x (in S). For every point $x \in S$, let $R(x) \stackrel{\text{def}}{=} \inf\{R \geq 0 : |\bar{B}_S(x, R)| \geq \delta |S|\}$. Since $|S| \geq 2$ is finite, $R(x)$ is defined and furthermore $|B_S(x, R(x))| < \delta |S|$. By our assumption, for all $x \in X$, $|B_S(x, 2R(x))| > (1-\delta)|S|$, and hence each annulus $A_S(x_i, R(x_i), 2R(x_i))$ contains at least $(1 - 2\delta)|S|$ points.

Let $x_0 \in S$ be the point for which $R(x_0)$ is minimal, and iteratively for $t = 1, 2, \ldots$ choose $x_t \in S$ to be an arbitrary point of

$$\bigcap_{i=0}^{t-1} A_S(x_i, R(x_i), 2R(x_i)).$$

Clearly we can continue this process as long as the above intersection remains non-empty. Suppose we are forced to stop after selecting k points $x_0, x_1, \ldots, x_{k-1}$. On the one hand, we threw away at most $2\delta|S|$ points at every step, and thus $k \geq 1/2\delta$. On the other hand, the set $U = \{x_0, x_1, \ldots, x_{k-1}\}$ is contained in $B(x_0, 2R(x_0))$. Furthermore, for any pair x_i, x_j with $i < j$, we see that $d(x_i, x_j) \geq R(x_i)$ since $x_j \notin B_S(x_i, R(x_i))$. But by construction, $R(x_i) \geq R(x_0)$ for all $i \geq 0$. It follows that the set U has aspect ratio at most 4, and thus by Lemma 1, $k \leq 2^{O(\dim(X))}$. We conclude that $\delta \geq 1/2k \geq (\frac{1}{2})^{O(\dim(X))}$.

The ring-separator tree. Given the above lemma, it is natural to define a *δ-ring-separator tree* for a metric space (X, d). This is a binary tree where each node has a label $S \subseteq X$, constructed recursively as follows. The root of the tree has the label $S = X$. A node labelled by S is a leaf if $|S| = 1$, and has two children if $|S| \geq 2$. In the latter case, we take (x, R) to be a δ-ring-separator of S, and add under the node an *inside child*, whose label is $S_I = B_S(x, 2R)$, and an *outside child*, whose label is $S_O = S \setminus B_S(x, R)$. Note that S_I and S_O are not a partition of S, as their intersection is generally non-empty. Let us add the ring-separator into the node's label and say that the non-leaf node's label is $\langle S, (x, R)\rangle$ (where $S \subseteq X$, $x \in S$ and $R > 0$). Lemma 3 shows that if $|S| \geq 2$ then S admits a δ-ring-separator with $\delta \geq (\frac{1}{2})^{O(\dim(X))}$. Since every step decreases the size of S by a factor of at least $1 - \delta$, the height of the tree is at most $2^{O(\dim(X))} \log n$.

The 3-ANN algorithm. We now show how to use ring-separator trees to solve the 3-ANN problem on X in time $2^{O(\dim(X))} \log n$. Unfortunately, a bound of $2^{O(\dim(X))} \log n$ on the height of the ring-separator tree implies a possibly huge space requirement of $n^{2^{O(\dim(X))}}$. This problem will be remedied in Section 3.2.

Let q be the query against X. The algorithm proceeds along a root to leaf path, i.e., starts at the root and recursively goes down the tree until a leaf node is met. Suppose that we are at a node $N = \langle S, (x, R)\rangle$. If $d(q, x) \leq 3R/2$, the algorithm proceeds to the inside child of N; otherwise, it proceeds to the outside child. Eventually, a leaf node $N = \langle\{x\}\rangle$ is met. Let x_i be the point x seen in the ith node along this root to leaf path (either the point from the ring-separator or the only point in S). Then the algorithm outputs the point which is closest to q among the points x_i.

This algorithm clearly runs in time linear in the height of the tree, i.e. $2^{O(\dim(X))} \log n$. We now proceed to show that the point x_i which is output is indeed a 3-approximate nearest neighbor to q.

Proposition 1. *The above algorithm outputs a 3-approximate nearest neighbor to q.*

Proof. Let $a^* \in X$ be the real nearest neighbor to q, i.e. $d(q,X) = d(q,a^*)$. Let $N_1, N_2, \ldots, N_k$ be the sequence of tree nodes seen by the algorithm on input q. For $i < k$ let $N_i = \langle S_i, (x_i, R_i)\rangle$, and let $N_k = \langle \{x_k\}\rangle$. Clearly $a^* \in S_1$ since $S_1 = X$. If $a^* \in S_k$, then $x_k = a^*$, and in this case the algorithm returns the exact nearest neighbor. Otherwise, there exists some j for which $a^* \in S_j$ but $a^* \notin S_{j+1}$. We claim that in this case, x_j is a 3-approximate nearest neighbor to q.

If N_{j+1} is the inside child of N_j, then $d(q,x_j) \le 3R_j/2$, yet $d(a^*, x_j) \ge 2R_j$, so by the triangle inequality,

$$d(q,a^*) \ge d(a^*,x_j) - d(q,x_j) \ge 2R_j - 3R_j/2 = R_j/2 \ge d(q,x_j)/3.$$

If N_{j+1} is the outside child of N_j, then $d(q,x_j) \ge 3R_j/2$, yet $d(a^*,x_j) \le R_j$. By the triangle inequality $d(q,a^*) \ge R_j/2$, and we conclude that

$$d(x_j,q) \le d(x_j,a^*) + d(a^*,q) \le R_j + d(a^*,q) \le 3\, d(a^*,q).$$

The proof follows by recalling that the algorithm outputs the closest point to q among $x_1, \ldots, x_k$.

3.2 Polynomial storage

We now discuss how to achieve a space requirement that is polynomial in n, regardless of $\dim(X)$, by modifying the ring-separator tree algorithm of Section 3.1. In a nutshell, we introduce three techniques that, when applied together, "massage" this decision tree into a polynomial size directed acyclic graph (DAG) that can be used for $O(1)$-ANN. First, we "canonicalize" the decision tree by snapping every δ-ring-separator to a suitable net of the metric. This step limits the number of distinct radii that are used by the ring-separators in the data structure. Second, we eliminate the need for outside children in the decision tree by using several (actually $2^{O(\dim(X))}\log n$) inside children. This step opens the possibility to take a path in the tree (sequence of inside children) that corresponds to properly nested balls and represent the information in the entire sequence by a single ball (namely, the last one). This modification is crucial for the third step, in which we "fold" the decision tree onto itself, by merging nodes that have the same role (i.e., correspond to the same ball). A crucial detail in this step is a subtle invariant that is maintained when going into smaller and smaller balls (inside children). Specifically, whenever we go into a ball $B_X(y, 2R)$, we know that $d(y,q) \le \gamma R$ for a suitable constant $1 < \gamma < 2$. This guarantees that every sequence of balls that we generate is indeed properly nested.

We sketch the resulting ring-separator DAG and algorithm, deferring details to the full version of the paper. Fix, for every R which is a power of 2, an R-net Y_R of X such that $Y_R \subseteq Y_{R/2}$. Each vertex of the DAG is a tuple $\langle y, R\rangle$, where $y \in Y_R$ and R is a power of 2, and represents the ball $S = B_X(y, 2R)$. (It is easy to bound the number of non-trivial values R per node y by $O(n)$.) If $S = B_X(y, 2R)$ contains only one point of X then the vertex has no outgoing edges. Otherwise, let (x,t) be a ring-separator for S. The first technique above guarantees that

$x \in Y_{t/2}$. We now add an outgoing edge to every vertex $\langle Y', R'\rangle$ where $R' \geq t$ is a power of 2 and $y' \in Y_{R'/32}$ such that $\frac{3}{4}R' \leq d(x, y') \leq 3R'$ and $d(y, y') \leq \frac{3}{2}R$. (The case $R' \leq t$ is slightly different.) Given a query point q, the $O(1)$-ANN algorithm traverses the DAG, starting from a fixed vertex corresponding to a ball that contains all of X, and ending when a sink is reached. When the algorithm is at the vertex $\langle y, R\rangle$ (assuming it is not a sink), we compute the value R' which is a power of 2 and $d(x, q) \in (R', 2R']$. (The case $d(x, q) \leq 2t$ is slightly different.) If there is an edge to a vertex $\langle y', R'/32\rangle$ with $d(y', q) \leq \frac{9}{8}R'/32$, the traversal proceeds along any such edge. Otherwise, the traversal stops. Eventually, the algorithm reports the closest point to the query among the points y and x seen along the traversal.

Acknowledgements

The authors would like to thank the Weizmann Institute, and in particular their host Uriel Feige.

References

[Ass83] P. Assouad. Plongements lipschitziens dans $\mathbf{R}^n$. *Bull. Soc. Math. France*, 111(4):429–448, 1983.

[Cla99] K. L. Clarkson. Nearest neighbor queries in metric spaces. *Discrete Comput. Geom.*, 22(1):63–93, 1999.

[GKL03] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *Proceedings of the 44th annual Symposium on the Foundations of Computer Science*, 2003.

[Gro99] M. Gromov. *Metric structures for Riemannian and non-Riemannian spaces.* Birkhäuser, Boston, 1999.

[HKMR04] K. Hildrum, J. Kubiatowicz, S. Ma, and S. Rao. A note on finding nearest neighbors in growth-restricted metrics. In *Proceedings of the 15th annual ACM-SIAM Symposium on Discrete Algorithms*, 2004.

[H01] S. Har-Peled. A replacement for Voronoi diagrams of near linear size. In *42nd IEEE Symposium on Foundations of Computer Science (Las Vegas, NV, 2001)*, pages 94–103. IEEE Computer Soc., Los Alamitos, CA, 2001.

[IM98] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *30th Annual ACM Symposium on Theory of Computing*, pages 604–613, May 1998.

[KKL03] S. Kakade, M. Kearns, and J. Langford. Exploration in metric state spaces. In *Proc. of the 20th International Conference on Machine Learning*, 2003.

[KL04] 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*, 2004.

[KOR98] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. In *30th Annual ACM Symposium on the Theory of Computing*, pages 614–623, 1998.

[KR02] D. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. In *34th Annual ACM Symposium on the Theory of Computing*, pages 63–66, 2002.

[Tal04] K. Talwar. Bypassing the embedding: Approximation schemes and distance labeling schemes for growth restricted metrics. To appear in the procedings of the *36th annual Symposium on the Theory of Computing*, 2004.