Beyond Locality-Sensitive Hashing

by

Ilya Razenshteyn

M.S., Moscow State University (2012)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

Author . . . .
Department of Electrical Engineering and Computer Science
May 21, 2014

Certified by . . . .
Piotr Indyk
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Chairman, Department Committee on Graduate Students

Beyond Locality-Sensitive Hashing

by

Ilya Razenshteyn

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Engineering

Abstract

We present a new data structure for the c-approximate near neighbor problem (ANN) in the Euclidean space. For n points in $\mathbb{R}^d$, our algorithm achieves $O_c(n^\rho + d \log n)$ query time and $O_c(n^{1+\rho} + d \log n)$ space, where $\rho \leq 0.73/c^2 + O(1/c^3) + o_c(1)$. This is the first improvement over the result by Andoni and Indyk (FOCS 2006) and the first data structure that bypasses a locality-sensitive hashing lower bound proved by O'Donnell, Wu and Zhou (ICS 2011). By known reductions we obtain a data structure for the Hamming space and ℓ_1 norm with $\rho \leq 0.73/c + O(1/c^{3/2}) + o_c(1)$, which is the first improvement over the result of Indyk and Motwani (STOC 1998).

Thesis Supervisor: Piotr Indyk
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

The biggest thank you goes to my research advisor Piotr Indyk. Piotr is a unique blend of a brilliant researcher and extremely friendly, supportive and passionate personality. When I first met Piotr in Moscow in December 2011, I could not dream that he would become my advisor. I am truly looking forward to working with Piotr during (at least) the rest of my PhD studies. Thank you Piotr!

Second, let me thank my undegraduate advisors Maxim Babenko and Alexander Shen for showing me beauty of Theoretical Computer Science.

This thesis is based on a paper [4] that is a joint work with Alex Andoni, Piotr Indyk and Nguyen Le Huy. I would like to thank Alex, Piotr and Huy for being excellent collaborators with very "hands-on" approach to working on problems. It was incredible pleasure to work with you.

Thanks to Akamai for their fellowhip that supported me during the first year of my studies at MIT. Thanks to Google Research for the internship during Summer 2013, and particularly to my mentors Silvio Lattanzi and Vahab Mirrokni.

I would like to thank my friends from MIT. I can not think of more comfortable and friendly environment. In particular, Theory Group changed my attitude to social life quite a lot. Let me specifically mention Artūrs Bačkurs, Rati Gelashvili, Gautam "G" Kamath, Sepideh Mahabadi and Adrian Vladu (Artūrs, Gautam and Sepideh also helped me immensely by selflessly proofreading the the first version of the paper this thesis is based on). I would like to thank Luke Schaeffer for allowing me to include his proof of Lemma 2.3.7.

Thanks to Silvio Micali for being a great academic advisor. Registration days with Silvio always leave me reflecting on what I would like to do after graduation. Thank you to administrative assistants Joanne Hanley, Linda Lynch, Nina Olff and Rebecca Yadegar for doing a great job fighting bureaucracy and paperwork.

Finally, I would like to express deep gratitude to my parents Lyudmila and Petr, my sister Eleonora, and my wife Oksana for their love.

Contents

Chapter 1

Introduction

The near neighbor search problem is defined as follows: given a set P of n points in a d-dimensional space, build a data structure that, given a query point q, reports any point within a given distance r to the query (if one exists). The problem is of major importance in several areas, such as data mining, information retrieval, computer vision, databases and signal processing.

Many efficient near(est) neighbor algorithms are known for the case when the dimension d is "low" (e.g., see [20], building on [7]). However, despite decades of effort, the current solutions suffer from either space or query time that are exponential in the dimension d. This phenomenon is often called "the curse of dimensionality". To overcome this state of affairs, several researchers proposed *approximation* algorithms for the problem. In the (c, r)-approximate near neighbor problem (ANN), the data structure is allowed to return any data point whose distance from the query is at most cr, for an approximation factor $c > 1$. Many approximation algorithms for the problem are known, offering tradeoffs between the approximation factor, the space and the query time. See [2] for an up to date survey.

From the practical perspective, the space used by an algorithm should be as close to linear as possible. If the space bound is (say) sub-quadratic, and the approximation factor c is a constant, the best existing solutions are based on *locality-sensitive hashing* [13, 12]. The idea of that approach is to hash the points in a way that the probability of collision is much higher for points which are close (with the distance

r) to each other than for those which are far apart (with distance at least cr). Given such hash functions, one can retrieve near neighbors by hashing the query point and retrieving elements stored in buckets containing that point. If the probability of collision is at least p_1 for the close points and at most p_2 for the far points, the algorithm solves the (c, r)-ANN using $n^{1+\rho+o(1)}$ extra space and $dn^{\rho+o(1)}$ query time[1], where $\rho = \log(1/p_1)/\log(1/p_2)$ [12]. The value of the exponent ρ depends on the distance function and the locality-sensitive hash functions used. In particular, it is possible to achieve $\rho = 1/c$ for the ℓ_1 norm [13], and $\rho = 1/c^2 + o_c(1)$ for the ℓ_2 norm [3].

It is known that the above bounds for the value of ρ are *tight*. Specifically, we have that, for all values of c, $\rho \geq 1/c - o_c(1)$ for the ℓ_1 norm[2] [22]. A straightforward reduction implies that $\rho \geq 1/c^2 - o_c(1)$ for the ℓ_2 norm. Thus, the running time of the simple LSH-based algorithm, which is determined by ρ, cannot be improved.

Results In this thesis we show that, despite the aforementioned limitation, the space and query time bounds for ANN can be substantially improved. In particular, for the ℓ_2 norm, we give an algorithm with query time $O_c(n^\eta + d \log n)$ and space $O_c(n^{1+\eta} + d \log n)$, where $\eta \leq 0.73/c^2 + O(1/c^3) + o_c(1)$ that gives an improvement for large enough c. This also implies an algorithm with the exponent $\eta \leq 0.73/c + O(1/c^{3/2}) + o_c(1)$ for the ℓ_1 norm, by a classic reduction from ℓ_1 to ℓ_2-squared [17]. These results constitute the first improvement to the complexity of the problem since the works of [13] and [3].

Techniques Perhaps surprisingly, our results are obtained by using essentially the same LSH functions families as described in [3] or [13]. However, the properties of those hash functions that we exploit, as well as the overall algorithm, are different. On a high-level, our algorithms are obtained by combining the following two observations:

1. After a slight modification, the existing LSH functions can yield better values

[1]Assuming that each hash function can be sampled and evaluated in $n^{o(1)}$ time, stored in $n^{o(1)}$ space, that distances can be computed in $O(d)$ time, and that $1/p_1 = n^{o(1)}$.

[2]Assuming $1/p_1 = n^{o(1)}$.

of the exponent ρ if the search radius r is comparable to the diameter[3] of the point-set. This is achieved by augmenting those functions with a "center point" around which the hashing is performed. See Section 1.1 for an intuition why this approach works, in the (somewhat simpler) context of the Hamming distance.

2. We can ensure that the diameter of the point-set is small by applying standard LSH functions to the original point-set P, and building a separate data structure for each bucket.

This approach leads to a two-level hashing algorithm. The *outer hash table* partitions the data sets into buckets of bounded diameter. Then, for each bucket, we build the *inner hash table*, which uses (after some pruning) the center of the minimum enclosing ball of the points in the bucket as a center point. Note that the resulting two-level hash functions cannot be "unwrapped" to yield a standard LSH family, as each bucket uses slightly different LSH functions, parametrized by different center points. That is, the two-level hashing is done in a *data-aware* manner while the standard LSH functions are chosen from a distribution independent from the data. This enables us to overcome the lower bound of [22].

Many or most of the practical applications of LSH involve designing data-aware hash functions ([32, 26, 31, 30, 19, 28], to name a few). Unfortunately, not many rigorous results in this area are known. The challenge of understanding and exploiting the relative strengths of data-oblivious versus data-aware methods has been recognized as a major open question in the area (e.g., see [1], page 77). Our results can be viewed as a step towards that goal.

Related work In this thesis we assume worst case input. If the input is generated at random, it is known that one can achieve better running times. Specifically, assume that all points are generated uniformly at random from $\{0,1\}^d$, and the query point is "planted" at distance $d/(2c)$ from its near neighbor. In this setting, the work of [6, 10, 16, 25] gives an exponent of $\frac{1}{\ln 4 \cdot c} \approx \frac{0.73}{c}$.

[3]In the analysis we use a notion that is weaker than the diameter. However, we ignore this detail for now for the sake of clarity.

Even better results are known for the problem of finding the *closest pair* of points in a dataset. In particular, the algorithm of [9] for the random closest pair has an exponent of $1 + \frac{1}{2c-1}$.[4] More recently, [29] showed how to obtain an algorithm with a runtime exponent < 1.79 for any approximation $c = 1 + \varepsilon$ in the random case. Moreover, [29] also gives an algorithm for the worst-case closest pair problem with a runtime exponent of $2 - \Omega(\sqrt{\varepsilon})$ for $c = 1 + \varepsilon$ approximation.

There are also two related lines of lower bounds for ANN. First, the work of [21] showed that LSH for Hamming space must have $\rho \geq 1/(2c) - O(1/c^2) - o_c(1)$, and [22] improved the lower bound to $\rho \geq 1/c - o_c(1)$. Second, [23, 24] have given cell-probe lower bounds for ℓ_1 and ℓ_2, roughly showing that any randomized ANN algorithm for the ℓ_1 norm must either use space $n^{1+\Omega(1/(tc))}$ or more than t cell-probes. We note that the LSH lower bound of $\rho \geq 1/(2c)$ from [21] might more naturally predict lower bounds for ANN because it induces a "hard distribution" that corresponds to the aforementioned "random case" . In contrast, if one tries to generalize the LSH lower bound of [22] into a near neighbor hard distribution, one obtains a dataset with special structure, which one can exploit (and our algorithm will indeed exploit such structure). In fact, the LSH lower bound of [21] has been used (at least implicitly) in the data structure lower bounds from [23, 24].

1.1 Intuition behind the improvement

We give a brief intuition on why near neighbor instances with bounded diameter are amenable to more efficient LSH functions. For simplicity we consider the Hamming distance as opposed to the Euclidean distance.

Assume that all input points, as well as the query point, are within the Hamming distance of s from each other. By shifting one of the data points to the origin, we can assume that all points have at most s non-zeros (i.e., ones). Consider any data point p and the query point q. To make calculations easier, we assume that both p and q

[4]Note that a near neighbor search algorithm with query time n^ρ and space/preprocessing time of $n^{1+\rho}$ naturally leads to a solution for the closest pair problem with the runtime of $n^{1+\rho}$.

have exactly s ones.

The "standard" LSH functions for the Hamming distance project the points on one of the coordinates selected uniformly at random. For two points p and q this results in a collision probability of $1 - \|p - q\|_1/d$, which is $1 - r/d$ and $1 - cr/d$ for points within the distance of r and cr, respectively. The probability gap of $1 - x$ vs. $1 - cx$ leads to the exponent ρ equal to $1/c$ [13]. To improve on this, we can instead use the min-wise hash functions of [5]. For those functions, the probability of collision between two points p and q is equal to $\frac{|p \cap q|}{|p \cup q|}$, where $\cup$ and $\cap$ denote the union and intersection of two Boolean vectors, respectively. Since we assumed that $\|p\|_1 = \|q\|_1 = s$, we have

$$\frac{|p \cap q|}{|p \cup q|} = \frac{\|p\|_1 + \|q\|_1 - \|p - q\|_1}{\|p\|_1 + \|q\|_1 + \|p - q\|_1} = \frac{2s - \|p - q\|_1}{2s + \|p - q\|_1} = \frac{1 - \|p - q\|_1/(2s)}{1 + \|p - q\|_1/(2s)}$$

As a result, the collision probability gap for distances r and cr becomes $\frac{1-x}{1+x}$ vs. $\frac{1-cx}{1+cx}$. This leads to ρ that is lower than $1/c$.

1.2 Roadmap

In Section 2.2 we show how to get an improved exponent for the case, when all the points and queries lie in a ball of radius $O(cr)$. We achieve this in three steps: we gradually move from a sphere to a spherical shell, and then to a ball. This Section culminates in Lemma 2.2.8.

In Section 2.3 we utilize the above "low-diameter" family and show how to solve the general case of ANN achieving the improved improved exponent. We carefully partition $\mathbb{R}^d$ in a hierarchical manner on different distance scales. The main technical result we obtain is Theorem 2.3.8. Then the data structure for ANN follows almost immediately (Theorem 2.3.9). The preprocessing time we obtain is quasi-quadratic, but we can reduce it to quasi-linear at the expense of increasing the exponent to

$$\frac{0.87}{c^2} + O\left(\frac{1}{c^3}\right) + o_c(1).$$

See the discussion in Section 2.3.8 for more details.

Chapter 2

Approximate Near Neighbor

2.1 Preliminaries

In the text we denote the ℓ_2 norm by $\|\cdot\|$. When we use $O(\cdot)$, $o(\cdot)$, $\Omega(\cdot)$ or $\omega(\cdot)$ we explicitly write all the parameters that the corresponding constant factors depend on as subscripts. Our main tool will be hash families $\mathcal{H}$ on a metric space. We can identify a hash function with a partition of the space. For a partition $\mathcal{P}$ and a point p we denote $\mathcal{P}(p)$ the corresponding part of $\mathcal{P}$. If $\mathcal{P}$ is a partial partition of a subset of the space, then we denote $\bigcup \mathcal{P}$ the union of all pieces of $\mathcal{P}$. By $N(a, \sigma^2)$ we denote a standard Gaussian with mean a and variance σ^2. We denote the closed ball with center $u \in \mathbb{R}^d$ and radius $r \geq 0$ by $B(u, r)$.

Definition 2.1.1. The (c, r)*-approximate near neighbor problem (ANN)* with failure probability f is to construct a data structure over a set of points P in metric space (X, D) supporting the following query: given any fixed query point $q \in X$, if there exists $p \in P$ with $D(p, q) \leq r$, then report some $p' \in P$ such that $D(p', q) \leq cr$, with probability at least $1 - f$.

Remark: note that we allow preprocessing to be randomized as well, and we measure the probability of success over the random coins tossed during *both* preprocessing and query phases.

Definition 2.1.2 ([12])**.** For a metric space (X, D) we call a family of hash functions

$\mathcal{H}$ on X (r_1, r_2, p_1, p_2)-*sensitive*, if for every $x, y \in X$ we have

- if $D(x, y) \leq r_1$, then $\Pr_{h \sim \mathcal{H}}[h(x) = h(y)] \geq p_1$;
- if $D(x, y) \geq r_2$, then $\Pr_{h \sim \mathcal{H}}[h(x) = h(y)] \leq p_2$.

Remark: for $\mathcal{H}$ to be useful we should have $r_1 < r_2$ and $p_1 > p_2$.

Definition 2.1.3. If $\mathcal{H}$ is a family of hash functions on a metric space X, then for any $k \in \mathbb{N}$ we can define a family of hash function $\mathcal{H}^{\otimes k}$ as follows: to sample a function from $\mathcal{H}^{\otimes k}$ we sample k functions $h_1, h_2, \ldots, h_k$ from $\mathcal{H}$ independently and map $x \in X$ to $(h_1(x), h_2(x), \ldots, h_k(x))$.

Lemma 2.1.4. *If $\mathcal{H}$ is (r_1, r_2, p_1, p_2)-sensitive, then $\mathcal{H}^{\otimes k}$ is (r_1, r_2, p_1^k, p_2^k)-sensitive.*

Theorem 2.1.5 ([12])**.** *Suppose there is a (r, cr, p_1, p_2)-sensitive family $\mathcal{H}$ for (X, D), where $p_1, p_2 \in (0, 1)$ and let $\rho = \ln(1/p_1)/\ln(1/p_2)$. Then there exists a data structure for (c, r)-ANN over a set $P \subseteq X$ of at most n points, such that:*

- *the query procedure requires at most $O(n^\rho/p_1)$ distance computations and at most $O(n^\rho/p_1 \cdot \lceil \log_{1/p_2} n \rceil)$ evaluations of the hash functions from $\mathcal{H}$ or other operations;*
- *the data structure uses at most $O(n^{1+\rho}/p_1)$ words of space, in addition to the space needed to store the set P.*

The failure probability of the data structure can be made to be arbitrarily small constant.

Remark: this theorem says that in order to construct a good data structure for the (c, r)-ANN it is sufficient to have a (r, cr, p_1, p_2)-sensitive family $\mathcal{H}$ with small $\rho = \ln(1/p_1)/\ln(1/p_2)$ and not too small p_1.

We use the LSH family crafted in [3]. The properties of this family that we need are summarized in the following theorem.

Theorem 2.1.6 ([3]). *Suppose that $d \to \infty$ and $c > 1$. There exists an LSH family $\mathcal{H}$ for $\mathbb{R}^d$ that is $(1, \beta c, p_1, p_2(\beta))$-sensitive for every $\beta \geq 1$, where $\widetilde{p_1} = \exp(-O(d^{1/4}))$ and*

$$\widetilde{\rho}(c, \beta) = \frac{\ln(1/p_1)}{\ln(1/p_2(\beta))} \leq \left(1 + O\left(\frac{\ln d}{d^{1/4}}\right)\right) \cdot \frac{1}{(\beta c)^2}.$$

Moreover, a function from $\mathcal{H}$ can be sampled in time $\exp(O(\sqrt{d} \ln d))$, stored in space $\exp(O(\sqrt{d} \ln d))$ and evaluated in time $\exp(O(\sqrt{d} \ln d))$.

We use Johnson-Lindenstrauss dimension reduction procedure.

Theorem 2.1.7 ([14], [8]). *For every $d \in \mathbb{N}$ and $\varepsilon, \delta > 0$ there exists a distribution over linear maps $A \colon \mathbb{R}^d \to \mathbb{R}^{O(\log(1/\delta)/\varepsilon^2)}$ such that for every $x \in \mathbb{R}^d$ one has $\mathrm{Pr}_A[\|Ax\| \in (1 \pm \varepsilon)\|x\|] \geq 1 - \delta$. Moreover, such a map can be sampled in time $O(d \log(1/\delta)/\varepsilon^2)$.*

Combining Theorem 2.1.5, Theorem 2.1.6, Theorem 2.1.7 one has the following corollary.

Corollary 2.1.8. *There exists a data structure for (c, r)-ANN for ℓ_2^d with preprocessing time and space $O_c(n^{1+1/c^2+o_c(1)} + nd)$ and query time $O_c(dn^{1/c^2+o_c(1)})$.*

Proof. Using THeorem 2.1.7, we can reduce d to $O_c(\log n)$ by increasing probability of error by an arbitrarily small constant and degrading in terms of c slightly. Then, we just plug Theorem 2.1.6 into Theorem 2.1.5. □

We use the following standard estimate on tails of Gaussians (see, e.g., [15]).

Lemma 2.1.9 ([15]). *For every $t > 0$*

$$\frac{1}{\sqrt{2\pi}} \cdot \left(\frac{1}{t} - \frac{1}{t^3}\right) \cdot e^{-t^2/2} \leq \mathrm{Pr}_{X \sim N(0,1)}[X \geq t] \leq \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{t} \cdot e^{-t^2/2}.$$

Finally, let us state Jung's theorem.

Theorem 2.1.10 (see, e.g., Exercise 1.3.5 in [18]). *Every subset of $\mathbb{R}^d$ with diameter Δ can be enclosed in a ball of radius $\Delta/\sqrt{2}$.*

2.2 The case of low diameter

In this section we build an LSH family for a ball of radius $O(c)$ with $\rho \approx (1-\Omega(1))/c^2$. To accomplish this, we proceed in three steps.

First, we show how to handle *a sphere* of radius $O(c)$. This requires a new partitioning procedure that is somewhat reminiscent of the one used by Karger et al [15] for approximate coloring of a graph. Second, we treat *a spherical shell* of radius $O(c)$ and width $O(1)$. We reduce this case to the case of sphere by normalizing lengths of points involved. Finally, we build an LSH family for the whole ball. The argument goes by cutting the ball into spherical shells of constant width. The second and the third steps are quite straightforward (albeit somewhat technical).

2.2.1 Sphere

In this section we show how to build a good LSH family $\mathcal{G}_1$ for a unit sphere $S^{d-1} \subset \mathbb{R}^d$. The construction is similar to the rounding scheme for SDP relaxation for graph coloring developed by Karger, Motwani and Sudan [15]. The partitioning process parametrized by a positive real number $\varepsilon > 0$ can be summarized by the following pseudocode (see Algorithm 1). In principle, the sampling process can be infinite,

Algorithm 1 Partitioning of a sphere

```
$\mathcal{P} \leftarrow \emptyset$
while $\bigcup \mathcal{P} \neq S^{d-1}$ do
    sample $w \sim N(0,1)^d$
    $S \leftarrow \left\{u \in S^{d-1} \mid \langle u, w\rangle \geq \varepsilon\sqrt{d}\right\} \setminus \bigcup \mathcal{P}$
    if $S \neq \emptyset$ then
        $\mathcal{P} \leftarrow \mathcal{P} \cup \{S\}$
    end if
end while
```

but for a moment, let us not worry about efficiency issues, and compute collision probabilities instead.

Let $u, v \in S^{d-1}$ be two points on the sphere with angle $\alpha = \angle(u, v)$ between them.

Let us understand the probability $\Pr_{g\sim\mathcal{G}_1}[g(u)=g(v)]$. If $\varepsilon\sqrt{d}\geq 2$, then we have

$$\begin{aligned}
&\Pr_{g\sim\mathcal{G}_1}[g(u)=g(v)]\\
&=\frac{\Pr_{w\sim N(0,1)^d}\left[\langle u,w\rangle\geq \eta c\cdot\varepsilon\sqrt{d}\text{ and }\langle v,w\rangle\geq \eta c\cdot\varepsilon\sqrt{d}\right]}{\Pr_{w\sim N(0,1)^d}\left[\langle u,w\rangle\geq \eta c\cdot\varepsilon\sqrt{d}\text{ or }\langle v,w\rangle\geq \eta c\cdot\varepsilon\sqrt{d}\right]}\\
&=\frac{\Pr_{X,Y\sim N(0,1)}\left[X\geq\varepsilon\sqrt{d}\text{ and }\cos\alpha\cdot X-\sin\alpha\cdot Y\geq\varepsilon\sqrt{d}\right]}{\Pr_{X,Y\sim N(0,1)}\left[X\geq\varepsilon\sqrt{d}\text{ or }\cos\alpha\cdot X-\sin\alpha\cdot Y\geq\varepsilon\sqrt{d}\right]}\\
&\in[1;2]\cdot\frac{\Pr_{X,Y\sim N(0,1)}\left[X\geq\varepsilon\sqrt{d}\text{ and }\cos\alpha\cdot X-\sin\alpha\cdot Y\geq\varepsilon\sqrt{d}\right]}{\Pr_{X\sim N(0,1)}\left[X\geq\varepsilon\sqrt{d}\right]}\\
&\subseteq\left[\sqrt{2\pi};\sqrt{15\pi}\right]\cdot\varepsilon\sqrt{d}\cdot\frac{\Pr_{X,Y\sim N(0,1)}\left[X\geq\varepsilon\sqrt{d}\text{ and }\cos\alpha\cdot X-\sin\alpha\cdot Y\geq\varepsilon\sqrt{d}\right]}{e^{-\varepsilon^2 d/2}},
\end{aligned}\tag{2.1}$$

where the second step is by spherical symmetry of Gaussians, and the last step is by Lemma 2.1.9 and $\varepsilon\sqrt{d}\geq 2$.

The next two claims give almost tight estimates on

$$\Pr_{X,Y\sim N(0,1)}\left[X\geq\varepsilon\sqrt{d}\text{ and }\cos\alpha\cdot X-\sin\alpha\cdot Y\geq\varepsilon\sqrt{d}\right].$$

Claim 2.2.1. *For every $\xi>0$ and $0\leq\alpha<\pi$ one has*

$$\Pr_{X,Y\sim N(0,1)}[X\geq\xi\ \textit{and}\ \cos\alpha\cdot X-\sin\alpha\cdot Y\geq\xi]\leq\frac{1}{\sqrt{2\pi}}\cdot\frac{e^{-\xi^2\cdot(1+\tan^2\frac{\alpha}{2})/2}}{\xi\cdot\sqrt{1+\tan^2\frac{\alpha}{2}}}.$$

Proof. We have

$$\begin{aligned}
&\Pr_{X,Y\sim N(0,1)}\left[X \ge \xi \text{ and } \cos\alpha \cdot X - \sin\alpha \cdot Y \ge \xi\right]\\
&\le \Pr_{X,Y\sim N(0,1)}\left[(1+\cos\alpha)\cdot X - \sin\alpha \cdot Y \ge 2\xi\right]\\
&= \Pr_{X\sim N(0,1)}\left[\sqrt{2(1+\cos\alpha)}\cdot X \ge 2\xi\right]\\
&= \Pr_{X\sim N(0,1)}\left[X \ge \xi \cdot \left(1+\tan^2\frac{\alpha}{2}\right)\right]\\
&\le \frac{1}{\sqrt{2\pi}} \cdot \frac{e^{-\xi^2\cdot(1+\tan^2\frac{\alpha}{2})/2}}{\xi\cdot\sqrt{1+\tan^2\frac{\alpha}{2}}},
\end{aligned}$$

where the second step is by 2-stability of Gaussians and the last step is due to Lemma 2.1.9. □

Claim 2.2.2. *For every $\xi > 0$ and $0 \le \alpha \le \alpha_0 \le \pi/2$ such that $\xi \cdot \tan\frac{\alpha_0}{2} \ge 2$ one has*

$$\Pr_{X,Y\sim N(0,1)}\left[X \ge \xi \textit{ and } \cos\alpha \cdot X - \sin\alpha\cdot Y \ge \xi\right] \ge \frac{1}{4\pi}\cdot\frac{e^{-\xi^2\cdot(1+\tan^2\frac{\alpha_0}{2})/2}}{\xi^2\cdot\tan\frac{\alpha_0}{2}}.$$

Proof. We have

$$\begin{aligned}
&\Pr_{X,Y\sim N(0,1)}\left[X \ge \xi \text{ and } \cos\alpha \cdot X - \sin\alpha \cdot Y \ge \xi\right]\\
&\ge \Pr_{X,Y\sim N(0,1)}\left[X \ge \xi \text{ and } Y \le -\xi\cdot\tan\tfrac{\alpha}{2}\right]\\
&= \Pr_{X\sim N(0,1)}\left[X\ge\xi\right]\cdot \Pr_{Y\sim N(0,1)}\left[Y \ge \xi\cdot\tan\frac{\alpha}{2}\right]\\
&\ge \Pr_{X\sim N(0,1)}\left[X\ge\xi\right]\cdot \Pr_{Y\sim N(0,1)}\left[Y \ge \xi\cdot\tan\frac{\alpha_0}{2}\right]\\
&\ge \frac{1}{4\pi}\cdot\frac{e^{-\xi^2\cdot(1+\tan^2\frac{\alpha_0}{2})/2}}{\xi^2\cdot\tan\frac{\alpha_0}{2}},
\end{aligned}$$

where the first step follows from $\alpha \le \pi/2$. Indeed, since in this case $\sin\alpha, \cos\alpha \ge 0$, we have

$$\cos\alpha\cdot X - \sin\alpha\cdot Y \ge \xi\cdot\left(\cos\alpha + \sin\alpha\cdot\tan\frac{\alpha}{2}\right) = \xi.$$

The last step follows from Lemma 2.1.9, the inequality $\xi\cdot\tan\frac{\alpha_0}{2} \ge 2$ and the bound $\alpha_0 \le \pi/2$ (the latter is used to conclude that $\xi \ge 2$ as well). □

Now combining (2.1), Claim 2.2.1 and Claim 2.2.2, we get the following estimates on collision probabilities.

Claim 2.2.3. *If $\varepsilon\sqrt{d} \geq 2$ and $0 \leq \alpha < \pi$ then*

$$\ln \frac{1}{\Pr_{g \sim \mathcal{G}_1}\left[h(u) = h(v)\right]} \geq \frac{\varepsilon^2 d \cdot \tan^2 \frac{\alpha}{2}}{2} + \frac{\ln\left(1 + \tan^2 \frac{\alpha}{2}\right)}{2} - \ln 3.$$

If $0 \leq \alpha \leq \alpha_0 \leq \pi/2$ and $\varepsilon\sqrt{d} \cdot \tan \frac{\alpha_0}{2} \geq 2$ then

$$\ln \frac{1}{\Pr_{g \sim \mathcal{G}_1}\left[h(u) = h(v)\right]} \leq \frac{\varepsilon^2 d \cdot \tan^2 \frac{\alpha_0}{2}}{2} + \ln\left(\varepsilon\sqrt{d} \cdot \tan \frac{\alpha_0}{2}\right) + \frac{\ln 8\pi}{2}.$$

The next Claim is essentially a restatement of the one above, but with low order terms being hidden. This version is more convenient to use.

Claim 2.2.4. *There exist absolute constants $\delta_0, C > 0$ such that for every $\varepsilon, \alpha_0, \delta > 0$, $d \in \mathbb{N}$ with $\delta \leq \delta_0$, $\alpha_0 \leq \pi/2$, $\varepsilon\sqrt{d} \cdot \tan \frac{\alpha_0}{2} \geq C \cdot \sqrt{\frac{\ln(1/\delta)}{\delta}}$ we have*

$$\ln \left(\inf_{\substack{u,v \in S^{d-1} \\ \angle(u,v) \leq \alpha_0}} \Pr_{g \sim \mathcal{G}_1}\left[g(u) = g(v)\right] \right)^{-1} \in (1 \pm \delta) \cdot \frac{\varepsilon^2 d \cdot \tan \frac{\alpha_0}{2}}{2}.$$

Also, for every α_1 with $\alpha_0 \leq \alpha_1 < \pi$ we have

$$\frac{\ln \left(\inf_{\substack{u,v \in S^{d-1} \\ \angle(u,v) \leq \alpha_0}} \Pr_{g \sim \mathcal{G}_1}\left[g(u) = g(v)\right] \right)^{-1}}{\ln \left(\sup_{\substack{u,v \in S^{d-1} \\ \angle(u,v) \geq \alpha_1}} \Pr_{g \sim \mathcal{G}_1}\left[g(u) = g(v)\right] \right)^{-1}} \leq (1 + \delta) \cdot \left(\frac{\tan \frac{\alpha_0}{2}}{\tan \frac{\alpha_1}{2}} \right)^2.$$

Finally, we conclude with a good LSH family for a sphere that is *efficient*. Basically, it is a modification of $\mathcal{G}_1$, but we stop sampling after not too many iterations.

Claim 2.2.5. *Suppose that $d \to \infty$, $1/2 \leq \eta \leq \eta_0$ for a constant η_0 and $c \geq c_0$ for an absolute constant $c_0 > 0$. There exists an LSH family $\mathcal{G}_2$ for a sphere of radius ηc*

that is $(1, \beta c, p_1, p_2(\beta))$*-sensitive for every* $\beta \geq 1$*, where* $p_1 = \exp(-\Theta_{\eta_0,c}(\sqrt{d}))$ *and*

$$\rho(\beta) = \frac{\ln(1/p_1)}{\ln(1/p_2(\beta))} \leq \left(1 + O_{\eta_0,c}\left(\frac{\ln d}{\sqrt{d}}\right)\right) \cdot F_2(\eta, c, \beta),$$

where

$$F_2(\eta, c, \beta) \leq \left(1 - \frac{\beta^2}{4\eta^2}\right) \cdot \frac{1}{(\beta c)^2} + O\left(\frac{1}{c^4}\right).$$

as $c \to \infty$*.*

Moreover, a function from this family can be sampled in time $\exp(O_{\eta_0,c}(\sqrt{d}))$*, stored in space* $\exp(O_{\eta_0,c}(\sqrt{d}))$ *and evaluated in time* $\exp(O_{\eta_0,c}(\sqrt{d}))$*.*

Proof. We instantiate $\mathcal{G}_1$ with $\varepsilon = d^{-1/4}$. Using the notation of Claim 2.2.4, we have

$$\tan^2 \frac{\alpha_0}{2} = \frac{1}{4\eta^2 c^2 - 1} \tag{2.2}$$

and

$$\tan^2 \frac{\alpha_1}{2} = \frac{\beta^2 c^2}{4\eta^2 c^2 - \beta^2 c^2}. \tag{2.3}$$

As a result, we have $\alpha_0 = \Omega_{\eta_0,c}(1)$, so, we can take δ from Claim 2.2.4 to be $O_{\eta_0,c}\left(\frac{\ln d}{\sqrt{d}}\right)$. Then, using (2.2), (2.3) and Claim 2.2.4, we conclude that we can achieve

$$p_1 = \exp(-\Theta_{\eta_0,c}(\sqrt{d}))$$

and

$$F_2(\eta, c, \beta) = \frac{4\eta^2 - \beta^2}{\beta^2(4\eta^2 c^2 - 1)} \leq \left(1 - \frac{\beta^2}{4\eta^2}\right) \cdot \frac{1}{(\beta c)^2} + O\left(\frac{1}{c^4}\right).$$

In the last step, $O(\cdot)$-notation should be understood as $c \to \infty$.

Now let us address the issue of efficiency. We modify $\mathcal{G}_1$ as follows. Instead of sampling vectors until we cover the whole sphere, we just sample T vectors and stop. Clearly, in order not to spoil guarantees that we have proved about $\mathcal{G}_1$ it is sufficient to choose T so that

$$\Pr\left[\text{we cover the whole sphere after } T \text{ samples}\right] \geq 1 - \exp(-\Omega_{\eta_0,c}(d)).$$

It is not hard to see using union bound over a sufficiently fine net and tail bounds for Gaussians and Chi-squared random variables that we can set $T = \exp(O_{\eta_0,c}(\sqrt{d}))$ (for $\varepsilon = d^{-1/4}$). □

2.2.2 Spherical shell

Now let us consider the case of a spherical shell: that is, we will be working with a set

$$\left\{u \in \mathbb{R}^d \mid \|u\| \in [\eta c - 1; \eta c + 1]\right\}$$

for a constant η. We reduce this case to the sperical case by normalizing the lengths of all vectors involved.

Claim 2.2.6. *If $u, v \in \mathbb{R}^d$, $\|u - v\| = d$ and $\|u/\|u\| - v/\|v\|\| = \widetilde{d}$, then*

$$\widetilde{d}^2 = \frac{d^2 - (\|u\| - \|v\|)^2}{\|u\|\|v\|}.$$

Claim 2.2.7. *Suppose that $d \to \infty$, $1/2 - 1/c \leq \eta \leq \eta_0 + 1/c$ for a constant η_0 and $c \geq c_0$ for an absolute constant $c_0 > 0$. There exists an LSH family $\mathcal{G}_3$ for a spherical shell with radii $\eta c - 1$ and $\eta c + 1$ that is $(1, \beta c, p_1, p_2(\beta))$-sensitive for every $\beta \geq 1$, where $p_1 = \exp(-\Theta_{\eta_0,c}(\sqrt{d}))$ and*

$$\rho(\beta) = \frac{\ln(1/p_1)}{\ln(1/p_2(\beta))} \leq \left(1 + O_{\eta_0,c}\left(\frac{\ln d}{\sqrt{d}}\right)\right) \cdot F_3(\eta, c, \beta),$$

where

$$F_3(\eta, c, \beta) \leq \left(1 - \frac{\beta^2}{4\eta^2}\right) \cdot \frac{1}{(\beta c)^2} + O\left(\frac{1}{c^3}\right).$$

as $c \to \infty$.

Moreover, a function from this family can be sampled in time $\exp(O_{\eta_0,c}(\sqrt{d}))$, stored in space $\exp(O_{\eta_0,c}(\sqrt{d}))$ and evaluated in time $\exp(O_{\eta_0,c}(\sqrt{d}))$.

Proof. We first normalize a point to lie on the sphere of radius ηc, and then we hash it using a family $\mathcal{G}_2$ from Claim 2.2.5.

Modifying the proof of Claim 2.2.5 slightly (using Claim 2.2.6), we have

$$\tan^2 \frac{\alpha_0}{2} \le \frac{1}{4(\eta c - 1)^2 - 1}$$

and

$$\tan^2 \frac{\alpha}{2} \ge \frac{\beta^2 c^2 - 4}{4(\eta c + 1)^2 - \beta^2 c^2 + 4}.$$

Overall, we can set

$$F_3(\eta, c, \beta) \le \frac{4(\eta c + 1)^2 - \beta^2 c^2 + 4}{(\beta^2 c^2 - 4)(4(\eta c - 1)^2 - 1)} \le \left(1 - \frac{\beta^2}{4\eta^2}\right) \cdot \frac{1}{(\beta c)^2} + O\left(\frac{1}{c^3}\right).$$

□

2.2.3 Ball

Finally, we are able to handle the case of a ball of radius ηc for a constant η.

Lemma 2.2.8. *Suppose that $d \to \infty$, $1/2 - 1/c \le \eta \le \eta_0$ for a constant η_0 and $c \ge c_0$ for an absolute constant $c_0 > 0$. There exists an LSH family $\mathcal{G}_4$ for a ball of radius $\eta c + 1$ that is $(1, \beta c, p_1, p_2(\beta))$-sensitive for every $\beta \ge 1$, where $p_1 = \exp(-O_{\eta_0, c}(d^{2/3}))$ and*

$$\rho(\beta) = \frac{\ln(1/p_1)}{\ln(1/p_2(\beta))} \le \left(1 + O_{\eta_0, c}\left(\frac{1}{d^{1/6}}\right)\right) \cdot F_4(\eta, c, \beta),$$

where

$$F_4(\eta, c, \beta) \le \left(1 - \frac{\beta^2}{4\eta^2}\right) \cdot \frac{1}{(\beta c)^2} + O\left(\frac{1}{c^3}\right).$$

as $c \to \infty$.

Moreover, a function from this family can be sampled in time $\exp(O_{\eta_0, c}(\sqrt{d}))$, stored in space $\exp(O_{\eta_0, c}(\sqrt{d}))$ and evaluated in time $\exp(O_{\eta_0, c}(\sqrt{d}))$.

Proof. The main idea is to discretize the ball and then apply Claim 2.2.7. The following pseudocode (Algorithm 2) shows how to partition a ball $B(0, \eta c + 1) \subseteq \mathbb{R}^d$.

The bounds on space and time for this family are immediate. Indeed, we have $O_{\eta_0, c}(1)$ "slices" and $k(s) = O_{\eta_0, c}(d^{1/6})$ for every slice (in line 20).

Algorithm 2 Partitioning of a ball

```
1: $\mathcal{P} \leftarrow \emptyset$
2: $\widetilde{p_1} \leftarrow \infty$
3: for $s \leftarrow 1 \ldots \lceil \eta c \rceil$ do
4:     if $2(s+1) \geq c$ then
5:         Let $p_1(s)$ be $p_1$ from Claim 2.2.7 for a shell with radii $s-1$ and $s+1$
6:         if $p_1(s) < \widetilde{p_1}$ then
7:             $\widetilde{p_1} \leftarrow p_1(s)$
8:         end if
9:     end if
10: end for
11: Let $\sigma$ be a random permutation of integers from 1 to $\lceil \eta c \rceil$
12: for $i \leftarrow 1 \ldots \lceil \eta c \rceil$ do
13:     $s \leftarrow \sigma(i)$
14:     $R(s) \leftarrow B(0, \eta c) \setminus \bigcup \mathcal{P}$
15:     if $R(s) \neq \emptyset$ then
16:         if $2(s+1) < c$ then
17:             $\mathcal{P} \leftarrow \mathcal{P} \cup \{R(s)\}$
18:         else
19:             Let $k(s)$ be the smallest positive integer such that $p_1(s)^{k(s)} \leq \widetilde{p_1}^{d^{1/6}}$
20:             Let $\mathcal{P}'$ be a partition of $R(s)$ according to $g \sim \mathcal{G}_3^{k(s)}$, where $\mathcal{G}_3$ is a
    family from Claim 2.2.7 for a shell with radii $s-1$ and $s+1$
21:             $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}'$
22:         end if
23:     end if
24: end for
```

Let us prove the bound on p_1. Suppose we have two points $u, v \in B(0, \eta c + 1)$ with $\|u - v\| \le 1$. Then, there exists s such that $s - 1 \le \|u\|, \|v\| \le s + 1$. With probability $\Omega(1)$ both u and v will lie in $R(s)$ (here we use that we sample a random permutation σ). Conditioned on this event, the probability of collision is at least $p_1(s)^{k(s)}$. Thus, overall,

$$p_1 \ge \Omega(1) \cdot \widetilde{p_1}^{d^{1/6}+1} = \exp(-O_{\eta_0,c}(d^{2/3})).$$

Moreover, for every s with $2(s+1) > c$ we have

$$p_1(s)^{k_1(s)} \in p_1^{1 \pm O_{\eta_0,c}(d^{-1/6})}. \tag{2.4}$$

Now let us prove the desired bound on $\rho(\beta)$. Suppose that $u, v \in B(0, \eta c)$ with $\|u - v\| \ge \beta c$, where $\beta \ge 1$. If u and v do not lie simultaneously in one $R(s)$, then we are done. Otherwise, suppose that $u, v \in R(s)$. Obviously, $2(s+1) \ge c$, so we partition $R(s)$ using Claim 2.2.7. From (2.4) and Claim 2.2.7, we have

$$\rho(\beta) \le \left(1 + O_{\eta_0,c}\left(\frac{1}{d^{1/6}}\right)\right) \cdot F_3(s/c, c, \beta).$$

Since $F_3(s/c, c, \beta) \le F_3(\eta, c, \beta)$, we indeed can take

$$F_4(\eta, c, \beta) = F_3(\eta, c, \beta) \le \left(1 - \frac{\beta^2}{4\eta^2}\right) \cdot \frac{1}{(\beta c)^2} + O\left(\frac{1}{c^3}\right).$$

□

2.3 The general case

We now describe our near neighbor data structure. It is composed of several independent data structures, where each one is a hierarchical hashing scheme, described next. We will conclude with proving our main theorem for ANN search.

First, we provide some very high-level intuition.

2.3.1 Intuition

The general approach can be seen as using LSH scheme composed of two levels: the hash function is $f = (f_h, f_g)$ chosen from two families $f_h \in \mathcal{H}^{\otimes k'}$, $f_g \in \mathcal{G}_4^{\otimes l}$ for some k', l, where $\mathcal{H}$ is the "ball carving LSH" from Theorem 2.1.6, and $\mathcal{G}_4$ is the "low-diameter LSH" from Lemma 2.2.8. In particular, the hash function $f_g(p)$ will depend on the bucket $f_h(p)$ and the other dataset points in the bucket $f_h(p)$. Intuitively, the "outer level" hash function f_h performs a partial standard LSH partitioning (with $\rho \approx 1/c^2$), but also has the role of improving the "geometry" of the points (namely, the points in a buckets roughly will have a bounded diameter). After an application of f_h, the pointset (inside a fixed bucket) has bounded diameter, allowing us to use the improved "low-diameter" partitioning ("inner level"), with $\rho < 1/c^2$.

In more detail, first, let us recall the main idea of the proof of Theorem 2.1.5. Suppose that $\mathcal{H}$ is, say, a family from Theorem 2.1.6. Then, we choose k to be an integer such that for every $p, q \in \mathbb{R}^d$ with $\|p - q\| \geq c$ we have

$$\Pr_{h \sim \mathcal{H}^{\otimes k}}[h(p) = h(q)] \approx n^{-1}. \tag{2.5}$$

Then, by Theorem 2.1.6, we have for every $p, q \in \mathbb{R}^d$ with $\|p - q\| \leq 1$

$$\Pr_{h \sim \mathcal{H}^{\otimes k}}[h(p) = h(q)] \approx n^{-1/c^2}.$$

Now suppose we hash all the points using a function $h \sim \mathcal{H}^{\otimes k}$. For a query q the average number of "outliers" in a bin that corresponds to q (points p such that $\|p - q\| > c$) is at most 1 due to (2.5). On the other hand, for a data point p such that $\|p - q\| \leq 1$ the probability of collision is at least n^{-1/c^2}. Thus, we can create n^{1/c^2} independent hash tables, and query them all in time around $O(n^{1/c^2})$. The resulting probability of success is constant.

The above analysis relies on two distance scales: 1 and c. To get a better algorithm for ANN we introduce the third scale: τc for $\tau > 1$ being a parameter. First, we hash all the points using $\mathcal{H}^{\otimes k'}$ (where $k' \approx k/\tau$) so that the collision probabilities are roughly as follows.

Distance	1	c	τc
Probability of collision	$n^{-1/(\tau c)^2}$	n^{-1/τ^2}	n^{-1}

Now we can argue that with high probability any bucket has diameter $O_\tau(c)$. This allows us to use the family from Lemma 2.2.8 for each bucket and set probabilities of collision as follows.

Distance	1	c
Probability of collision	$n^{-(1-\Omega_\tau(1))\cdot(1-1/\tau^2)/c^2}$	n^{-1+1/τ^2}

Due to the independence, we expect overall collision probabilities to be products of "outer" collision probabilities from the first table and "inner" probabilities from the second table. Thus, in total, we have the following probabilities.

Distance	1	c
Probability of collision	$n^{-(1-\Omega_\tau(1))/c^2}$	n^{-1}

Then we argue as before and achieve

$$\rho \approx \frac{1-\Omega_\tau(1)}{c^2}.$$

This plan is not quite rigorous for several reasons. One of them is we do not properly take care of conditioning on the event "all buckets have low diameter". Nevertheless, in this section we show how to analyze a similar scheme rigorously. Also, in the actual construction we use more than two levels.

2.3.2 Construction

Suppose we want to solve $(c, 1)$-ANN in $\mathbb{R}^d$ for an n-point subset P. First, by applying Johnson–Lindenstrauss Lemma (Theorem 2.1.7) we can assume that $d = \Theta_c(\log n)$. Thus, all quantities of order $\exp(o(d))$ are $n^{o_c(1)}$.

Our data structure consists of a hierarchy of partitions of $\mathbb{R}^d$. Let l be a positive integer parameter. We build partitions $\mathcal{P}_0$, $\mathcal{P}_1$, …, $\mathcal{P}_{l-1}$, where $\mathcal{P}_i$ is a refinement of

$\mathcal{P}_{i-1}$. Also, we prune our pointset. Namely, $P_0 = P$, then, for every i, before building $\mathcal{P}_i$ we prune P_{i-1} and obtain P_i. We want the following conditions to hold:

- for every $p \in P_{l-1}$ and $q \in \mathbb{R}^d \setminus B(p, c)$ we have

$$\Pr\left[\mathcal{P}_{l-1}(p) = \mathcal{P}_{l-1}(q)\right] \le \frac{1}{n}; \tag{2.6}$$

- for every $p \in P$ and $q \in B(p, 1)$ we have

$$\Pr\left[p \in P_{l-1}, \mathcal{P}_{l-1}(p) = \mathcal{P}_{l-1}(q)\right] \ge \frac{1}{n^{\rho_0}}, \tag{2.7}$$

where $\rho_0 \approx 0.73/c^2$.

Let us present pseudocode for partitioning/pruning (Algorithm 3). We denote $\mathcal{H}$, $\widetilde{p_1}$ and $\widetilde{\rho}(c, \beta)$ according to Theorem 2.1.6. We use $\mathcal{G}_4(\eta)$, $p_1(\eta)$ and $\rho(\eta, c, \beta)$ according to Lemma 2.2.8. The partitioning process uses distance scales $\tau_1 c$, ..., $\tau_{l-1} c$, where $\tau_1 > \tau_2 > \ldots > \tau_l = 1$, which we will show how to choose later. Also, it will be crucial to choose "tensoring powers" $\widetilde{k}$, k_1, ..., k_{l-1} carefully.

Note that we can always find a ball required by line 12. This follows from Theorem 2.1.10.

2.3.3 Analysis

For the sake of uniformity we define $\mathcal{P}_{-1} = \left\{\mathbb{R}^d\right\}$ being a trivial partition.

To analyze the partitioning process let us introduce the following notation:

- for $0 \le i \le l - 1$ and $\beta \ge 1$ we denote

$$U(i, \beta) = \sup_{\substack{p \in P_i \\ q \in \mathbb{R}^d \setminus B(p, \beta c)}} \Pr\left[\mathcal{P}_i(p) = \mathcal{P}_i(q)\right];$$

- for $0 \le i \le l - 1$ denote

$$L(i) = \inf_{\substack{p \in P \\ q \in B(p, 1)}} \prod_{j=0}^{i} \Pr\left[\mathcal{P}_j(p) = \mathcal{P}_j(q) \mid \mathcal{P}_{j-1}(p) = \mathcal{P}_{j-1}(q), p \in P_j\right].$$

Algorithm 3 Hierarchical partitioning of $\mathbb{R}^d$

```
1: Let $\mathcal{P}_0$ be the partition of $\mathbb{R}^d$ induced by $h \sim \mathcal{H}^{\otimes \widetilde{k}}$
2: $P_0 \leftarrow P$
3: for $i \leftarrow 1 \dots l-1$ do
4:     $\mathcal{P}_i \leftarrow \emptyset$
5:     $P_i \leftarrow \emptyset$
6:     for $R$ is a part of $\mathcal{P}_{i-1}$ do
7:         $P' \leftarrow \{p \in P_{i-1} \mid p \in R\}$
8:         while $\exists p, p' \in P'$ with $\|p - p'\| > \tau_i c$ do
9:             $P' \leftarrow P' \setminus \{p, p'\}$
10:        end while
11:        if $P' \neq \emptyset$ then
12:            Let $u$ be the center of a ball of radius $\tau_i c/\sqrt{2}$ that covers $P'$
13:            Let $\mathcal{P}'$ be a partition of $B(u, \tau_i c/\sqrt{2}+1) \cap R$ according to $\mathcal{G}_4(\tau_i/\sqrt{2})^{k_i}$
14:            $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \mathcal{P}'$
15:            if $R \setminus \bigcup \mathcal{P}' \neq \emptyset$ then
16:                $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{R \setminus \bigcup \mathcal{P}'\}$
17:            end if
18:            $P_i \leftarrow P_i \cup P'$
19:        else
20:            $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{R\}$
21:        end if
22:    end for
23: end for
24: return $\mathcal{P}_{l-1}$
```

In words, $U(i, \beta)$ is an upper bound for the probability that a point $p \in P_i$ collides in $\mathcal{P}_i$ with a point $q \in \mathbb{R}^d$ such that $\|p - q\| \geq \beta c$. Similarly, $L(i)$ is a lower bound for the product of collision probabilities at different levels for points $p \in P$ and $q \in \mathbb{R}^d$ such that $\|p - q\| \leq 1$.

We can upper bound $U(\cdot, \cdot)$ and lower bound $L(\cdot)$ pretty easily as follows.

Claim 2.3.1.

$$U(i, \beta) \leq \widetilde{p_1}^{\widetilde{k}/\widetilde{\rho}(c,\beta)} \cdot \prod_{j=1}^{i} p_1(\tau_j/\sqrt{2})^{k_j/\rho(\tau_j/\sqrt{2},c,\beta)}$$

Proof. We have

$$\begin{aligned}
U(i, \beta) &= \sup_{\substack{p \in P \\ q \in \mathbb{R}^d \setminus B(p, \beta c)}} \Pr\left[p \in P_i, \mathcal{P}_i(p) = \mathcal{P}_i(q)\right] \\
&= \sup_{\substack{p \in P \\ q \in \mathbb{R}^d \setminus B(p, \beta c)}} \Pr\left[\forall\, 0 \leq j \leq i \;\; p \in P_j, \mathcal{P}_j(p) = \mathcal{P}_j(q)\right] \\
&\leq \sup_{\substack{p \in P \\ q \in \mathbb{R}^d \setminus B(p, \beta c)}} \prod_{j=0}^{i} \Pr\left[\mathcal{P}_j(p) = \mathcal{P}_j(q) \mid \mathcal{P}_{j-1}(p) = \mathcal{P}_{j-1}(q), p \in P_j\right] \\
&\leq \widetilde{p_1}^{\widetilde{k}/\widetilde{\rho}(c,\beta)} \cdot \prod_{j=1}^{i} p_1(\tau_j/\sqrt{2})^{k_j/\rho(\tau_j/\sqrt{2},c,\beta)},
\end{aligned}$$

where the third step is by "Markov property" of our partitioning process, and the last step is by Theorem 2.1.6 and Lemma 2.2.8. □

Now let us lower bound $L(\cdot)$.

Claim 2.3.2.

$$L(i) \geq \widetilde{p_1}^{\widetilde{k}} \cdot \prod_{j=1}^{i} p_1(\tau_j/\sqrt{2})^{k_j}$$

Proof. This is a straightforward implication of Theorem 2.1.6 and Lemma 2.2.8. □

Definition 2.3.3. We denote $U'(i, \beta)$ the upper bound on $U(i, \beta)$ from Claim 2.3.1 and $L'(i)$ the lower bound on $L(i)$ from Claim 2.3.2.

Now it is time to choose $\widetilde{k}$, k_1, …, k_{l-1}. We choose the powers so that we have $n \cdot U'(i, \tau_{i+1}) \leq L'(i+1)/2^{i+1}$ for every $0 \leq i \leq l-2$. Namely, we first fix $\widetilde{k}$ to be the smallest positive integer so that $n \cdot U'(0, \tau_1) \leq L'(1)/2$. Then, we similarly fix k_1 to achieve $n \cdot U'(1, \tau_2) \leq L'(2)/4$ etc. Finally, we fix the smallest possible k_{l-1} such that $n \cdot U'(l-1, 1) \leq 1$.

Now we can immediately claim (2.6).

Claim 2.3.4. *For every $p \in P_{l-1}$ and $q \in \mathbb{R}^d \setminus B(p, c)$ we have*

$$\Pr\left[\mathcal{P}_{l-1}(p) = \mathcal{P}_{l-1}(q)\right] \leq \frac{1}{n}.$$

Proof. This follows from the definition of $U(\cdot, \cdot)$, Claim 2.3.1 and the condition $n \cdot U'(l-1, 1) \leq 1$. □

Now we relate (2.7) and $L'(l-1)$. For this we need the following key result.

Claim 2.3.5. *For every $p \in P$ and $1 \leq i \leq l-1$ we have*

$$\Pr\left[p \notin P_i \mid p \in P_{i-1}\right] \leq n \cdot U'(i-1, \tau_i).$$

Proof. If p was not pruned until step $i-1$, and then got pruned and did not make it into P_i, the only reason for this could be that there exists $p' \in P_{i-1} \cap B(p, \tau_i c)$ such that $\mathcal{P}_{i-1}(p) = \mathcal{P}_{i-1}(p')$. Then by union bound, the definition of $U(\cdot, \cdot)$ and Claim 2.3.1 we get the result. □

Suppose that $p \in P$ and $q \in B(p, 1)$.

Claim 2.3.6. *For every $0 \leq i \leq l-1$*

$$\Pr\left[p \in P_i, \mathcal{P}_i(p) = \mathcal{P}_i(q)\right] \geq \frac{L'(i)}{2^i}.$$

Proof. The proof is by induction on i. For $i = 0$ we have

$$\Pr\left[p \in P_0, \mathcal{P}_0(p) = \mathcal{P}_0(q)\right] = \Pr\left[\mathcal{P}_0(p) = \mathcal{P}_0(q)\right] \geq L(0) \geq L'(0).$$

Now assume that $i > 0$. Then

$$\begin{aligned}
&\Pr\left[p \in P_i, \mathcal{P}_i(p) = \mathcal{P}_i(q)\right] \\
&= \Pr\left[p \in P_i, \mathcal{P}_i(p) = \mathcal{P}_i(q), p \in P_{i-1}, \mathcal{P}_{i-1}(p) = \mathcal{P}_{i-1}(q)\right] \\
&\geq \Pr\left[\mathcal{P}_i(p) = \mathcal{P}_i(q) \mid p \in P_{i-1}, \mathcal{P}_{i-1}(p) = \mathcal{P}_{i-1}(q)\right] \\
&\cdot \Pr\left[p \in P_{i-1}, \mathcal{P}_{i-1}(p) = \mathcal{P}_{i-1}(q)\right] - \Pr\left[p \notin P_i \mid p \in P_{i-1}\right] \\
&\geq \frac{L'(i)}{2^{i-1}} - n \cdot U'(i-1, \tau_i) \geq \frac{L'(i)}{2^i},
\end{aligned}$$

where the third step follows from the assumption of induction, the definition of $L(i)$, Claim 2.3.2 and Claim 2.3.5. The last step follows from the choice of powers $\widetilde{k}$ and k_j. $\square$

2.3.4 Computing ρ_0

In order to get the value ρ_0 in (2.7) by Claim 2.3.6 it is sufficient to lower bound $L'(l-1)$. To recall,

$$L'(i) = \widetilde{p_1}^{\widetilde{k}} \cdot \prod_{j=1}^{i} p_1(\tau_j/\sqrt{2})^{k_j}$$

and

$$U'(i, \beta) = \widetilde{p_1}^{\widetilde{k}/\widetilde{\rho}(c,\beta)} \cdot \prod_{j=1}^{i} p_1(\tau_j/\sqrt{2})^{k_j/\rho(\tau_j/\sqrt{2},c,\beta)}.$$

We choose $\widetilde{k}$ to be the smallest integer such that $n \cdot U'(0, \tau_1) \leq L'(1)/2$, then we choose k_1 to be the smallest integer such that $n \cdot U'(1, \tau_2) \leq L'(2)/4$ etc. Finally, we choose k_{l-1} to be the smallest integer such that $n \cdot U'(l-1, 1) \leq 1$. Let us denote α_0, α_1, …, α_{l-1} the following real numbers:

$$n^{-\alpha_i} = \begin{cases} \widetilde{p_1}^{\widetilde{k}}, & \text{if } i = 0, \\ p_1(\tau_i/\sqrt{2})^{k_i}, & \text{otherwise.} \end{cases}$$

The moment of reflection on the definitions of $L'(\cdot)$, $U'(\cdot,\cdot)$, on the choice of $\widetilde{k}$ and k_j, and on Theorem 2.1.6 and Lemma 2.2.8 reveals that

$$\alpha_i = \left(1 + O_{\tau_1,c,l}\left(\frac{1}{d^{1/6}}\right) + O_{\tau_1,l}\left(\frac{1}{c}\right)\right) \cdot \frac{\beta_i}{c^2},$$

where β_i is the solution of the following linear system: for every $0 \le i \le l-1$ we have

$$\tau_{i+1}^2 \beta_0 + \sum_{j=1}^{i} \frac{2\tau_{i+1}^2 \tau_j^2 \beta_j}{2\tau_j^2 - \tau_{i+1}^2} = 1.$$

So, using Claim 2.3.6, we get that in (2.7) we can achieve

$$\rho_0 = \left(1 + O_{\tau_1,c,l}\left(\frac{1}{d^{1/6}}\right) + O_{\tau_1,l}\left(\frac{1}{c}\right)\right) \cdot \frac{\sum_{i=0}^{l-1} \beta_i}{c^2},$$

where $\beta_0 = \frac{1}{\tau_1^2}$ and for every $1 \le i \le l-1$

$$\beta_i = \frac{2\tau_i^2 - \tau_{i+1}^2}{2\tau_i^2 \tau_{i+1}^2} \cdot \left(1 - \frac{\tau_{i+1}^2}{\tau_1^2} - \sum_{j=1}^{i-1} \frac{2\tau_{i+1}^2 \tau_j^2 \beta_j}{2\tau_j^2 - \tau_{i+1}^2}\right). \tag{2.8}$$

2.3.5 Choosing distance scales

The problem of getting a good ρ_0 in (2.7) is thus reduced to the following optimization question. For a given l choose $\tau_1 > \tau_2 > \ldots > \tau_l = 1$ so to minimize $\sum_{i=0}^{l-1} \beta_i$ subject to (2.8). For a general l this is a challenging optimization problem, but let us try to see what we can get for some interesting cases. First, if $l = 1$, then there is nothing to solve. We immediately get $\beta_0 = 1$. The first non-trivial case is $l = 2$, which corresponds to a two-level partitioning. In this case, we have one variable τ_1 to optimize over, and from (2.8) we get

$$\beta_0 + \beta_1 = \frac{1}{\tau_1^2} + \frac{2\tau_1^2 - 1}{2\tau_1^2} \cdot \left(1 - \frac{1}{\tau_1^2}\right).$$

It is immediate to see that $\beta_0 + \beta_1$ is minimized, when $\tau_1 = \sqrt{2}$, and in this case we have $\beta_0 + \beta_1 = \frac{7}{8}$. Then, for $l \ge 3$ the complexity of the optimization problem goes up very quickly. There are we are only able to obtain numerical results, which are summarized in the following table.

l	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	$\sum_{i=0}^{l-1}\beta_i$
1	1							1
2	$\sqrt{2}$	1						$7/8 = 0.875$
3	1.76	1.22	1					0.828
4	2.07	1.43	1.15	1				0.803
5	2.35	1.61	1.30	1.12	1			0.787
6	2.61	1.79	1.43	1.23	1.09	1		0.777
7	2.86	1.95	1.56	1.34	1.19	1.08	1	0.769

In the same time, since we do not really care about a particular value of l (as long as it is constant), we can consider a very large l and choose τ_i's suboptimally. Let us choose $\tau_i = \sqrt{l/i}$. In this case we get $\beta_i = \gamma_i/l$, where $\gamma_0 = 1$ and for every $i \geq 1$ we have

$$\gamma_i = \frac{i+2}{2} \cdot \left(\frac{i}{i+1} - \sum_{j=1}^{i-1} \frac{2\gamma_j}{2i-j+2} \right).$$

Observe that this sequence does not depend on l.

One can compute γ_i for quite large values of i and observe that it converges pretty rapidly to a real number between 0.72 and 0.73 (and so does $\frac{1}{l} \cdot \sum_{i=0}^{l-1} \gamma_i$). In particular, for $l = 40$ we get $\sum_{i=0}^{l-1} \beta_i < 0.73$ with this choice of thresholds. Next we prove that $\lim_{i\to\infty} \gamma_i = \frac{1}{\ln 4} \approx 0.721$ (which implies $\sum_{i=0}^{l-1} \beta_i \to \frac{1}{\ln 4}$). The proof is due to Luke Schaeffer [27].

Lemma 2.3.7.

$$\lim_{i\to\infty} \gamma_i = \frac{1}{\ln 4}$$

Proof. We have for every $i \geq 0$

$$\sum_{j=0}^{i} \frac{2\gamma_j}{2i-j+2} = 1.$$

Let us substitute $2\gamma_i = \frac{1}{\ln 2} + \varepsilon_i$. We need to show that $\lim_{i\to\infty} \varepsilon_i = 0$, where for every $i \geq 0$

$$\frac{H_{2i+2} - H_{i+1}}{\ln 2} + \sum_{j=0}^{i} \frac{\varepsilon_j}{2i - j + 2} = 1, \tag{2.9}$$

where H_n is the n-th harmonic number. It is known that

$$H_n = \ln n + \gamma + \frac{1}{2n} + O\left(\frac{1}{n^2}\right), \tag{2.10}$$

where γ is Euler's constant. Substituting (2.10) into (2.9), we get

$$\sum_{j=0}^{i} \frac{\varepsilon_j}{2i - j + 2} = \frac{1}{4i \ln 2} + O\left(\frac{1}{i^2}\right).$$

Thus,

$$\frac{1}{4 \ln 2} + O\left(\frac{1}{i}\right) \leq \sum_{j=0}^{i} \varepsilon_j \leq \frac{1}{2 \ln 2} + O\left(\frac{1}{i}\right). \tag{2.11}$$

Now we subtract (2.9) from itself for i and $i-1$:

$$\frac{H_{2i+2} + H_i - H_{i+1} - H_{2i}}{\ln 2} + \frac{\varepsilon_i}{i+2} + \sum_{j=0}^{i-1} \varepsilon_j \left(\frac{1}{2i - j + 2} - \frac{1}{2i - j}\right) = 0. \tag{2.12}$$

By (2.11), the third term of (2.12) is $O(1/i^2)$. The first term of (2.12) is

$$\frac{H_{2i+2} + H_i - H_{i+1} - H_{2i}}{\ln 2} = \frac{1}{\ln 2} \cdot \left(\frac{1}{2i+2} + \frac{1}{2i+1} - \frac{1}{i+1}\right) = O\left(\frac{1}{i^2}\right).$$

But this means that the second term of (2.12) is also $O(1/i^2)$. Thus, $\varepsilon_i = O(1/i)$. $\square$

One can ask: how optimal are thresholds $\tau_i = \sqrt{l/i}$? It turns out that if $l \to \infty$, then these thresholds are indeed optimal. This follows from three simple observations. First, $\sum_{i=0}^{l-1} \beta_i$ is monotone with respect to *adding* thresholds (adding threshold can not possibly hurt). Second, the sum is *continuous* as a function of τ_i's. Third, for any subset of real numbers larger than 1, we can approximate them arbitrarily well simultaneously with numbers $\sqrt{l/i}$ for some l. We omit proofs here, since this optimality is not needed for our main result.

2.3.6 Performance

In this section we analyze the performance of Algorithm 3. To recall, we assume that $d = \Theta_c(\log n)$ by Johnson-Lindenstrauss Lemma (Theorem 2.1.7).

We take $l = 40$ and by the results of Section 2.3.5, we can achieve

$$\rho_0 \le \frac{0.73}{c^2} + O\left(\frac{1}{c^3}\right) + o_c(1)$$

in (2.7).

It is easy to see that lines 1 and 13 of Algorithm 3 are executed $O(n)$ times, so overall space the whole partition can be stored in is $n^{1+o_c(1)}$ (by Theorem 2.1.6 and Lemma 2.2.8). Preprocessing time can be easily made $n^{2+o_c(1)}$. Indeed, lines 1 and 13 take time $n^{1+o_c(1)}$ in total, line 12 takes time $n^{1+o_c(1)}$ by an algorithm from [11]. So, the bottleneck is the pruning in lines 8–10, which can be easily done in time $n^{2+o_c(1)}$.

Finally, we can observe that it is possible to locate a query point $q \in \mathbb{R}^d$ in the partition $\mathcal{P}_{l-1}$ and, moreover, enumerate all the points $p \in P_{l-1}$ such that $\mathcal{P}_{l-1}(p) = \mathcal{P}_{l-1}(q)$ in time $n^{o_c(1)}$ plus constant time per point reported. For this, we store the hierarchy using a trie, where in each node we have a hash table.

We summarize this discussion in the following Theorem.

Theorem 2.3.8. *Suppose that $c > 1$. Let $P \subseteq \mathbb{R}^d$ be a pointset with $|P| = n$ and $d = \Theta_c(\log n)$. Then, in time $n^{2+o_c(1)}$ we can build a random partition $\mathcal{P}$ of $\mathbb{R}^d$ and a subset $P' \subseteq P$ such that for every $q \in \mathbb{R}^d$:*

- *for every $p \in \mathbb{R}^d \setminus B(q, c)$ we have*

$$\Pr\left[p \in P', \mathcal{P}(p) = \mathcal{P}(q)\right] \le \frac{1}{n};$$

- *for every $p \in B(q, 1)$ we have*

$$\Pr\left[p \in P', \mathcal{P}(p) = \mathcal{P}(q)\right] \ge \frac{1}{n^{\rho_0}},$$

where

$$\rho_0 \leq \frac{0.73}{c^2} + O\left(\frac{1}{c^3}\right) + o_c(1).$$

Moreover, in the same time we build a data structure that can be used to locate a point $q \in \mathbb{R}^d$ in $\mathcal{P}$ and report all the points $p \in P'$ such that $\mathcal{P}(p) = \mathcal{P}(q)$ in time $n^{o_c(1)}$ plus constant per point reported. The data structure takes $n^{1+o_c(1)}$ space to store.

2.3.7 The resulting data structure

To get from Theorem 2.3.8 to a data structure for $(c, 1)$-ANN for ℓ_2^d, we essentially repeat the argument from [12].

Theorem 2.3.9. *There exists a data structure for $(c, 1)$-ANN for ℓ_2^d with*

- *preprocessing time $O_c(n^{2+\rho_0} + nd \log n)$,*
- *space $O_c(n^{1+\rho_0} + d \log n)$,*
- *query time $O_c(n^{\rho_0} + d \log n)$,*

where

$$\rho_0 \leq \frac{0.73}{c^2} + O\left(\frac{1}{c^3}\right) + o_c(1).$$

Proof. The dimension reduction step takes time $O_c(nd \log n)$ by Theorem 2.1.7. After that we can apply Theorem 2.3.8. In order to make the probability of success constant we build and query $1/Q$ independent copies of the data structure from Theorem 2.3.8, where $Q = n^{\rho_0}$ and ρ_0 is the value from Theorem 2.3.8.

Note that the above bound on the query time is *in expectation*, but it is also possible to modify the algorithm slightly to get a worst-case bound. The algorithm still iterates over $1/Q$ data structures but stops after looking at $3/Q + 1$ points without finding an approximate near neighbor. The expected number of points the algorithm has to look at that are not an approximate near neighbor is at most $1/Q$. By Markov's inequality, with probability at least $2/3$, the algorithm doesn't look at more than $3/Q$ points that are not an approximate near neighbor. In each data structure, the probability that the algorithm fails to find an approximate near neighbor is at most $1-Q$. Thus, the probability it fails in all Q tables is at most $(1-Q)^{1/Q} \le 1/e$. Overall, with probability at least $1 - 1/3 - 1/e$, the algorithm finds an approximate near neighbor without looking at more than $3/Q + 1$ points. □

2.3.8 Quasi-linear time preprocessing

In Theorem 2.3.8 the preprocessing time is $n^{2+\rho_0+o_c(1)}$. The reason is the pruning procedure in Algorithm 3 (lines 8–10). We can get rid of it at the expense of increasing ρ_0. Namely, we do not do the expensive pruning, and instead we choose an arbitrary point from P' as a center, and cut off all the points that are further than $\tau_i c$ from it. Then, we proceed as before, but sample a function from $\mathcal{G}_4(\tau_i)$ instead of $\mathcal{G}_4(\tau_i/\sqrt{2})$.

The analysis remains the same, except Claim 2.3.1 and Claim 2.3.2, where we replace $\tau_i/\sqrt{2}$ with τ_i everywhere. Thus, (2.8) changes to

$$\beta_i = \frac{4\tau_i^2 - \tau_{i+1}^2}{4\tau_i^2\tau_{i+1}^2} \cdot \left(1 - \frac{\tau_{i+1}^2}{\tau_1^2} - \sum_{j=1}^{i-1} \frac{4\tau_{i+1}^2\tau_j^2\beta_j}{4\tau_j^2 - \tau_{i+1}^2}\right)$$

and choosing $\tau_i = \sqrt{l/i}$ as before, we get $\sum_{i=0}^{l-1}\beta_i = \frac{1}{l}\sum_{i=0}^{l-1}\gamma_i$, where $\gamma_0 = 1$ and

$$\gamma_i = \frac{i+4}{4} \cdot \left(\frac{i}{i+1} - \sum_{j=1}^{i-1} \frac{4\gamma_j}{4i - j + 4}\right).$$

This sequence seemingly converges to a real number between 0.86 and 0.87, and the following Lemma can be proved similarly to Lemma 2.3.7.

Lemma 2.3.10.

$$\lim_{i\to\infty} \gamma_i = \frac{1}{4\ln\frac{4}{3}}$$

(and thus

$$\sum_{i=0}^{l-1}\beta_i \to \frac{1}{4\ln\frac{4}{3}} \approx 0.869)$$

For instance, taking $l = 146$ we obtain

$$\rho_0 \leq \frac{0.87}{c^2} + O\left(\frac{1}{c^3}\right) + o_c(1).$$

Chapter 3

Conclusions and open problems

In this thesis we showed that data-dependent space partitioning can be beneficial for the Approximate Near Neighbor data structures.

One can ask the following natural question: **can one improve our bounds on the exponent?** We strongly suspect that the answer is positive. The first step might be to look at the following random instance. We consider a sphere of radius $cr/\sqrt{2}$ and sample n points on it at random. This way we obtain our dataset P. To generate a query, we choose a random point $p \in P$ and plant a point in $B(p, r)$. It can be shown that Lemma 2.2.8 gives the exponent around $1/(2c^2)$ for this case. It might be natural to try to improve it.

The second immediate question is **what are the limitations of data-dependent partitioning?** On one extreme, we can compute Voronoi diagram of P, which has exponential complexity, but partitions $\mathbb{R}^d$ perfectly. What are the other possibilities?

The third question is **can we perform instance-by-instance analysis of some partitioning scheme?** For example, it would be great to introduce a parameter of an instance and prove that the exponent ρ_0 we can achieve in (2.7) depends on this parameter.

Bibliography

[1] *Frontiers in Massive Data Analysis.* The National Academies Press, 2013.

[2] Alexandr Andoni. *Nearest Neighbor Search: the Old, the New, and the Impossible.* PhD thesis, Massachusetts Institute of Technology, 2009.

[3] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2006)*, pages 459–468, 2006.

[4] Alexandr Andoni, Piotr Indyk, Huy L. Nguyen, and Ilya Razenshteyn. Beyond locality-sensitive hashing. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '2014)*, pages 1018–1028, 2014.

[5] Andrei Z. Broder. Filtering near-duplicate documents. *Proceedings of the 1st International Conference on Fun with Algorithms (FUN '1998)*, 1998.

[6] Andrea Califano and Isidore Rigoutsos. FLASH: a fast look-up algorithm for string homology. In *Proceedings of the 1st International Conference on Intelligent Systems for Molecular Biology (ISMB '1993)*, pages 56–64, 1993.

[7] Kenneth L. Clarkson. A randomized algorithm for closest-point queries. *SIAM Journal on Computing*, 17(4):830–847, 1988.

[8] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures and Algorithms*, 22(1):60–65, 2003.

[9] Moshe Dubiner. Bucketing coding and information theory for the statistical highdimensional nearest-neighbor problem. *IEEE Transactions on Information Theory*, 56(8):4166–4179, 2010.

[10] Daniel H. Greene, Michal Parnas, and F. Frances Yao. Multi-index hashing for information retrieval. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science (FOCS '1994)*, pages 722–731, 1994.

[11] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization.* Springer-Verlag, 1988.

[12] Sariel Har-Peled, Piotr Indyk, and Rajeev Motwani. Approximate nearest neighbor: towards removing the curse of dimensionality. *Theory of Computing*, 8(1):321–350, 2012.

[13] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the 30th ACM Symposium on the Theory of Computing (STOC '1998)*, pages 604–613, 1998.

[14] William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Connecticut, 1982)*, volume 26 of *Contemporary Mathematics*, pages 189–206. 1984.

[15] David R. Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM*, 45(2):246–265, 1998.

[16] Richard M. Karp, Orli Waarts, and Geoffrey Zweig. The bit vector intersection problem (preliminary version). In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS '1995)*, pages 621–630, 1995.

[17] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[18] Jiří Matoušek. *Lectures on Discrete Geometry.* Springer, 2002.

[19] James McNames. A fast nearest-neighbor algorithm based on a principal axis search tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):964–976, 2001.

[20] Stefan Meiser. Point location in arrangements of hyperplanes. *Information and Computation*, 106(2):286–303, 1993.

[21] Rajeev Motwani, Rina Panigrahy, and Assaf Naor. Lower bounds on locality sensitive hashing. *SIAM Journal on Discrete Mathematics*, 21(4):930–935, 2007.

[22] Ryan O'Donnell, Yi Wu, and Yuan Zhou. Optimal lower bounds for locality sensitive hashing (except when q is tiny). In *Proceedings of Innovations in Computer Science (ICS '2011)*, pages 275–283, 2011.

[23] Rina Panigrahy, Kunal Talwar, and Udi Wieder. A geometric approach to lower bounds for approximate near-neighbor search and partial match. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2008)*, pages 414–423, 2008.

[24] Rina Panigrahy, Kunal Talwar, and Udi Wieder. Lower bounds on near neighbor search via metric expansion. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS '2010)*, pages 805–814, 2010.

[25] Ramamohan Paturi, Sanguthevar Rajasekaran, and John H. Reif. The light bulb problem. *Information and Computation*, 117(2):187–192, 1995.

[26] Ruslan Salakhutdinov and Geoffrey E. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

[27] Luke Schaeffer. Personal communication, 2014.

[28] Robert F. Sproull. Refinements to nearest-neighbor searching in k-dimensional trees. *Algorithmica*, 6(1-6):579–589, 1991.

[29] Gregory Valiant. Finding correlations in subquadratic time, with applications to learning parities and juntas. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS '2012)*, pages 11–20, 2012.

[30] Nakul Verma, Samory Kpotufe, and Sanjoy Dasgupta. Which spatial partition trees are adaptive to intrinsic dimension? In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '2009)*, pages 565–574, 2009.

[31] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Proceedings of 22nd Annual Conference on Neural Information Processing Systems (NIPS '2008)*, pages 1753–1760, 2008.

[32] Jay Yagnik, Dennis Strelow, David A. Ross, and Ruei-Sung Lin. The power of comparative reasoning. In *Proceedings of 13th IEEE International Conference on Computer Vision (ICCV '2011)*, pages 2431–2438, 2011.