INDIAN INSTITUTE OF SCIENCE

Bengaluru

E0-234: Introduction to Randomized Algorithms

Locality-Preserving Hashing in Multidimensional Spaces

Submitted To:
Dr. Arindam Khan
Asst. Professor
(Department of Computer Science & Automation)

Submitted By :
Kedarnath P
SR No: 20902
M.Tech CSA
(First year)

Contents

Abstract

We present a comprehensive literature survey on a paper titled *"Locality-Preserving Hashing in Multidimensional Spaces"* that investigates locality-preserving hashing for d-dimensional cubes, with applications in high-dimensional search and multimedia indexing. We discuss the key findings and results of the paper, including the effectiveness of simple and natural classes of hash functions for this problem. Our report aims to provide a clear and thorough understanding of the paper's methods, results, and implications, along with potential future research directions in the field of locality-preserving hashing.

1 Introduction

Hash functions play a crucial role in various applications, including near-neighbor retrieval and information retrieval. In a recent paper, Linial and Sasson [1] presented a groundbreaking theorem on hash functions that exhibits an intriguing property: points close to each other in the domain are hashed to points close to each other in the range. This theorem paves the way for new techniques in efficient near-neighbor retrieval, particularly in higher dimensions, which is an area of growing importance.

Theorem 1: There exists a family $\mathcal{G}$ of functions from an integer line $[1, \ldots, U]$ to $[1, \ldots, R]$ and a constant C such that for any $S \subset [1, \ldots, U]$ with $|S| \leq C\sqrt{R}$:

- $\Pr_{f \in \mathcal{G}} \left(f_{\{S\}} \text{ is one to one } \right) \geq \frac{1}{2}$
- all $f \in \mathcal{G}$ are non-expansive, i.e., for any $p, q \in U$, $d(f(p), f(q)) \leq d(p, q)$

The family $\mathcal{G}$ contains $O(|U|)$ functions, each of which is computable in $O(1)$ operations.

The above theorem, as proved by Linial and Sasson [1], states the existence of a family of hash functions, denoted as $\mathcal{G}$, that maps points from an integer line $[1, \ldots, U]$ to another integer line $[1, \ldots, R]$. This family of hash functions has two key properties: (1) a high probability of being one-to-one for a subset S with size less than or equal to $C\sqrt{R}$, and (2) non-expansiveness, which ensures that the distance between hashed points is no greater than the distance between the original points in the domain. Furthermore, the family $\mathcal{G}$ is computationally efficient, containing $O(|U|)$ functions that can be computed in $O(1)$ operations.

One potential application of this theorem is to improve near-neighbor retrieval in one-dimensional space, as demonstrated by Linial and Sasson [1]. By hashing a query point q to $h(q)$, one can efficiently search for the nearest k points within a specified distance δ in the domain. This locality-preserving property of the hash functions allows for good paging performance, as the neighborhood of the query point in the domain is not scattered across the range. This is contrary to Knuth's suggestion that tree search or digital tree search should be employed in virtual memory environments [2].

While there are various methods for retrieving near neighbors in one dimension, finding efficient techniques for higher dimensions remains challenging. Near-neighbor retrieval in higher dimensions is critical in a range of applications, such as information retrieval, pattern recognition [3, 4], statistics and data analysis [5, 6], machine learning [7], data compression [8], data mining [9], and image analysis [10].

Now, we explore near-neighbor retrieval in higher dimensions, specifically focusing on text and image retrieval. In text retrieval, vector-space methods [11, 12] are employed to represent documents as points in high-dimensional space. Dimensionality reduction techniques, such as principal components analysis [13], latent semantic indexing [14], and the Karhunen-Loéve/Hotelling transform [15, 16], can be applied to simplify the representation, but the number of dimensions may still remain high.

Similarly, image and multimedia retrieval involves extracting numerically-valued features from documents, such as color histograms, shape descriptors, and texture measurements [17, 18]. These features are represented as points in a multidimensional space, and dimensionality reduction techniques can be applied. User-defined queries are then transformed into the same vector space, and the retrieval task becomes one of finding near neighbors.

Current retrieval systems typically rely on brute-force linear search, calculating the distance between the query and all points in the database. However, no known high-dimensional search system has sublinear worst-case query performance. Generalizing Linial and Sasson's approach to higher dimensions offers a promising solution to this problem.

The authors present a generalized construction of locality-preserving hash functions for higher dimensions, supported by negative results suggesting that their construction is theoretically optimal. While their approach may only work for modest dimension values (e.g., 10-20), it provides a simple solution for indexing problems with moderate dimensions. This report also provides an overview of alternative near-neighbor retrieval methods to contextualize the authors' approach.

1.1 Other Existing Approaches for Nearest-Neighbor Search.

Author	Algorithm Time Complexity
Dobkin and Lipton[19]	Query time: $O(2^d \log n)$; Pre-processing: $O(n^{2^d})$
Clarkson [20]:	Query time: $O(\exp(d) \log n)$; Pre-processing: $O(n^{\lceil \frac{d}{2} \rceil (1+\epsilon)})$
Meiser [21]	Query time: $O(d^5 \log n)$; Pre-processing: $O(n^{d+\epsilon})$
Kleinberg [22]	Query time: $O(d^2 \log n)$; Pre-processing: $n^{O(d)}$

- The above table shows that different authors have used various data structures for near-neighbor search, which includes variants of k-d trees, R-trees,etc.
- Note that, the preprocessing time generally includes all of the necessary steps to prepare the data for efficient querying, which may include data cleaning, normalization, feature selection, dimensionality reduction, etc.
- Drawbacks: Computationally expensive for larger d .

The paper discusses various approaches to near-neighbor search in higher dimensions. Samet [23] surveys numerous data structures used for this problem, including variants of k-d trees, R-trees, and structures based on space-filling curves. While these structures perform reasonably well in 2 or 3 dimensions, their performance is generally poor in worst-case scenarios and typical cases in higher dimensions.

Computational geometry has contributed significantly to the study of proximity problems. Dobkin and Lipton [19] were among the first to establish upper bounds on the time required to answer a nearest-neighbor query in $\mathbb{R}^d$, providing an algorithm with query time $O(2^d \log n)$ and pre-processing $O(n^{2^d})$. Clarkson [20] improved upon this with an algorithm featuring query time $O(\exp(d) \cdot \log n)$ and pre-processing $O(n^{\lceil \frac{d}{2} \rceil (1+\epsilon)})$. Here, $\exp(d)$ denotes a function that grows at least as quickly as 2^d. Meiser [21] further improved query time to $O(d^5 \log n)$ with pre-processing $O\left(n^{d+\epsilon}\right)$. More recently, Kleinberg [22] developed a scheme for the approximate nearest neighbor problem, achieving query time $O(d^2 \log n)$ with preprocessing $n^{O(d)}$.

Despite these advancements, several other approaches and extensions (e.g., [24, 25, 26, 27, 28] have been proposed. However, the best approaches from these studies remain impractical for the values of d encountered in text and image retrieval applications discussed earlier .

1.2 Preliminaries

In this section, we explore the domain of hash functions considered in the paper, focusing on a d-dimensional cube $D = \{-U, \dots, U\}^d$. The range of these functions is a set of points in a d-dimensional cube I with side R. The paper presents results for the case when $d = 2$, and generalizations to higher dimensions. Although the study considers only cubes for the domain, the results can be extended to cuboids with unequal sides. Let $d(p, q)$ denote the distance between any points p and q, and we define $p = (x_p, y_p)$ and $q = (x_q, y_q)$. We can then define:

- $d_x(p, q) = |x_q - x_p|$ and $d_y(p, q) = |y_q - y_p|$
- $d_r(p, q) = (d_x(p, q)^r + d_y(p, q)^r)^{\frac{1}{r}}$, for any $r \geq 1$.

These definitions can also be extended to dimensions greater than 2.

Definition 1: A function $h : D \to I$ is defined as c-expansive under a distance metric d if for any $p, q \in D$, $d(h(p), h(q)) \leq d(p, q) + c$. If $c = 0$, then h is said to be non-expansive under d. If $c = 1$, then h is said to be 1-expansive under d and 2-expansive under d for $c = 2$.

The 1-dimensional family of hash functions $\mathcal{G}$ introduced by Linial and Sasson [1] is utilized in the constructions. These functions are obtained by "folding" the domain D along randomly chosen turning points in a way that any segment of D between two consecutive turning points has length $\Theta(R)$. In addition to the properties mentioned in the introduction, $\mathcal{G}$ has the property that $\Pr_{g \in \mathcal{G}}(g(x) = g(y)) = O(1/R)$ for any $x \neq y \in D$.

In the following sections, the subscript of $\Pr_{f \in \mathcal{F}}$ is often omitted, which denotes the probability space from which f is chosen. It should be assumed that f is chosen uniformly at random from the probability space implicit from the context.

2 Hashing in Two Dimensions

The main theorem in this section demonstrates the possibility of achieving $O(1)$ bucket size using locality-preserving hash functions in the 2-dimensional setting. To accomplish this, a family of 1-expansive hash functions, $\mathcal{H}_2 \subset h : D \to I$, is described. These functions are based on applying random rotations to the 2-dimensional domain cube I.

- Formally, $\mathcal{H}_2$ is defined as all functions h of the form $h(p) = t_2(g(r(p)))$, where for $p = (x_p, y_p)$,

- $t_c(p) = (\lfloor x_p/c \rfloor, \lfloor y_p/c \rfloor)$, and $g(p) = (g_x(x_p), g_y(y_p))$, where $g_x, g_y \in \mathcal{G}$;
- To define $r(p)$, let p_v be the column vector whose coordinates are the coordinates of p. Let $q_v = (x'_p, y'_p) = Mp_v$, where M is a 2×2 rotation matrix with column sums equal to 1. Then, $r(p) = (\lfloor x'_p \rfloor, \lfloor y'_p \rfloor)$, giving the rounded-off rotated coordinates.

The function r rotates the domain cube and rounds off the rotated points to the nearest lattice points.

Theorem 2: There exist constants C and B such that for any domain size U and range size R, the functions $h \in \mathcal{H}_2$ are 1-expansive under the d_1 distance metric and have the property that for any $p \in I$ and any $S \subset D$ with $|S| \leq CR$:

$$\Pr\left(\left|h^{-1}(p) \cap S\right| \leq B\right) > \frac{1}{2}$$

Theorem 2 states that there are specific constants, C and B, for a family of hash functions (h in $\mathcal{H}_2$). These hash functions have two main properties:

1. 1-expansive under the d_1 distance metric: This means that when these hash functions transform points, they increase the distances between them by a factor of at least 1. This helps in preserving the relative distances between points.
2. Controlled collisions: For any point in the index set (p) and a subset of the domain (S), with the size of S not exceeding C times the range size R, the probability of having a limited number of collisions (less than or equal to B) between the pre-image of p under the hash function and the subset S is greater than 1/2. In other words, the hash function does not map too many points from the subset S to the same hash value p with a probability greater than 50%.

Here, the intersection $(h^{-1}(p) \cap S)$ represents collisions. It refers to the set of points in the domain S that are mapped to the same hash value p. The theorem states that the number of these collisions is limited and controlled, making the hash function suitable for various applications where maintaining relative distances and minimizing collisions is desired.

The proof of Theorem 2 is obtained using Lemma 1, Lemma 2, and Lemma 4.

Lemma 1: Each $h \in \mathcal{H}_2$ is 1-expansive.

proof: It is easy to verify that each function r is 2-expansive, hence so is $g \circ r$. Application of t_2 reduces the expansion to 1.

For example: let

$$p = \begin{bmatrix} 7.0001 \\ 6.0001 \end{bmatrix}$$

and

$$q = \begin{bmatrix} 6.999 \\ 5.999 \end{bmatrix}$$

p, q are nearly very close points. Assume $Mp = p$ and $Mq = q$ after rotation and just before rounding off. Now, After applying the rounding-off,

$$r(p) = \begin{bmatrix} 7 \\ 6 \end{bmatrix}$$

and

$$r(q) = \begin{bmatrix} 6 \\ 5 \end{bmatrix}$$

Hence, $d_1(r(p), r(q)) \leq d_1(p, q) + 2$, here $c = 2$. As $g \in G$ and g is a non-expansive function, hence $g \circ r$ is also 2-expansive. In the t_2 reduction function, we divide by c and take the floor to make it 1-expansive under d_1 distance metric by choosing a suitable c.

Implication: The difference between the distance of hashed functions and the original points is at most 1.

Lemma 2: There exists a constant B_1 such that for each $p \in D$ and any $h \in \mathcal{H}2$, the number of $q \in D$ such that $d_\infty(p, q) < R$ and $h(p) = h(q)$ is less than B_1.

Proof: The proof follows from the construction of g, and the observation that for any $p \in D$, the number of $q \in D$ such that $r(p) = r(q)$ is bounded by a constant. To illustrate this, consider a basic example: Let after multiplying with the rotation matrix M,

$$Mp = \begin{bmatrix} 4.5 \\ 6.9 \end{bmatrix}$$

$$Mq = \begin{bmatrix} 4.8 \\ 6.4 \end{bmatrix}$$

After rounding-off, the hashed values of p, q are same and hence there is a collision.

r(p) = $\begin{bmatrix} 4 \\ 6 \end{bmatrix}$

r(q) = $\begin{bmatrix} 4 \\ 6 \end{bmatrix}$

Since there exists some constant B_1 as an upper bound for the number of collisions, this guarantees that the number of collisions is bounded when points are closer to each other.

Implication: Lemma 2 ensures that the number of points close to each other (i.e., $d_\infty(p, q) < R$) and having the same hash value is bounded by a constant B_1. This further supports the claim that the bucket size is $O(1)$.

Lemma 3: Let $p, q \in D$ be such that $d_\infty(p, q) \geq R$, then there is constant A such that $r(p), r(q)$ differ in both coordinates with probability at least $1 - A/R$.

Proof: In this proof for Lemma 3, the goal is to demonstrate that if p and q are two points in the domain D such that their infinity norm distance $d_\infty(p, q) \geq R$, then their rotated and rounded versions, $r(p)$ and $r(q)$, will differ in both coordinates with probability at least $1 - A/R$.

- Define a 2×2 matrix M as follows:

  $$M = \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \end{bmatrix},$$

  with the condition that $u_1 + u_2 = v_1 + v_2 = 1$. This matrix will be used to rotate the points p and q.

- Consider two points p and q such that their infinity norm distance, $d_\infty(p, q) = |x_p - x_q| \geq R$. This means that the points p and q are at least R units apart in the x-coordinate.

- Compute the difference between the x and y components of the rotated points $r(p)$ and $r(q)$. The difference in the x-components is given by $u_1(x_p - x_q) + v_1(y_p - y_q)$, and the difference in the y-components is given by $u_2(x_p - x_q) + v_2(y_p - y_q)$.

- The proof claims that there exists a constant A such that, with probability at least $(1 - A/R)$, both of the above expressions have a magnitude of at least 2. This means that, after rotation, the x and y components of the points $r(p)$ and $r(q)$ will still have a significant difference with high probability.

- After the rounding step, the points $r(p)$ and $r(q)$ will remain different in both coordinates with probability at least $1 - A/R$. This completes the proof of Lemma 3.

The proof relies on the existence of a constant A that ensures a significant difference in both coordinates after rotation and rounding.

Implication: As R increases, $1 - \frac{A}{R}$ increases and hence, we can say that the Rotation function helps to separate the points that are far apart very well.

The following fact which follows from the analysis due to Linial and Sasson [1].
Fact 1 There is a constant C_1 such that for any $p, q \in D$ if p and q differ on k coordinates, then

$$\Pr\left(t_2(g(p)) = t_2(g(q))\right) \le \frac{C_1}{R^k}$$

Lemma 4: There exists a constant C_2 such that for any $p, q \in D, d_\infty(p, q) \ge R$,

$$\Pr(h(p) = h(q)) \le \frac{C_2}{R^2}$$

Proof: We proceed as follows:

$$\Pr(h(p) = h(q)) \le \Pr\left(t_2(g(r(p))) = t_2(g(r(q)))\right) \tag{1}$$

Next, we consider two cases: 1) $r(p)$ and $r(q)$ differ in both coordinates, and 2) they differ in two coordinates. Combining these cases, we find the desired upper bound on hash collision probability:

$$\Pr(h(p) = h(q)) \le \frac{A}{R} \cdot \frac{C_1}{R} + \Pr\left(t_2(g(r(p))) = t_2(g(r(q))) | r(p), r(q) \text{ differ in 2 coordinates }\right) \quad \le \frac{C_2}{R^2}. \tag{2}$$

From Lemma 3, we know that if the points p and q are far apart, then the probability of their rotated points $r(p)$ and $r(q)$ differing in both coordinates is at least $1 - \frac{A}{R}$. In other words, the probability that $r(p)$ and $r(q)$ don't differ in both coordinates is at most $\frac{A}{R}$.

In the case where $r(p)$ and $r(q)$ don't differ in both coordinates, the probability of a hash collision is less than or equal to the product of the maximum probability that they don't differ in both coordinates, which is $\frac{A}{R}$, and the probability of a hash collision given by Fact 1, which is $\frac{C_1}{R}$ for one differing coordinate. Therefore, the probability of a hash collision in this case is bounded by $\frac{A}{R} \cdot \frac{C_1}{R}$.

Combining both cases, we get the desired upper bound on the probability of a hash collision for far away points

Implication: As the range R size increases, Lemma 4 states that the probability of number of collisions is low for far-away points, ensuring the hashing scheme effectively separates such far-away points.

3 Hashing in Higher Dimensions

In this section, the authors extend the locality-preserving hash families to higher-dimensional spaces. Instead of using the rotation matrices from the previous sections, they employ a more conventional approach, requiring the columns of the rotation matrix M to be orthonormal. This preserves the d_2 distance between points and allows them to obtain results for locality-preserving hashing under the d_2 distance metric.

We define $\mathcal{H}_d$ as the collection of all functions h of the form $h(p) = g(r(p))$, where for $p = (x_1, x_2, \ldots, x_d)$,

- $g(p) = (g_{x_1}(x_1), \ldots, g_{x_d}(x_d))$, where $g_{x_1}, \ldots, g_{x_d} \in \mathcal{G}$
- We define $q_v = (x'_1, \ldots, x'_d) = Mp_v$ where M is a $d \times d$ rotation matrix with orthonormal columns. Then, $r(p) = (\lfloor x'_1 \rfloor, \ldots, \lfloor y'_1 \rfloor)$

Theorem 3: There exist constants C and B such that for any dimension d, domain size U, and range size R, the functions $h \in \mathcal{H}_d$ are $\sqrt{d}$-expansive under the d_2 distance metric and have the property that for any $p \in I$ and any $S \subset D$ with $|S| \leq CR^{d/2}$,

$$\Pr\left(\left|h^{-1}(p) \cap S\right| \leq B^d\right) > \frac{1}{2}$$

Theorem 3 states that there are specific constants, C and B, for a family of hash functions (h in $\mathcal{H}_d$). These hash functions have two main properties:

- $\sqrt{d}$-expansive under the d_2 distance metric: This means that when these hash functions transform points, they increase the distances between them by a factor of at least $\sqrt{d}$. This helps in preserving the relative distances between points in higher dimensions.
- Controlled collisions: For any point in the index set (p) and a subset of the domain (S), with the size of S not exceeding C times the range size R raised to the power of $d/2$, the probability of having a limited number of collisions (less than or equal to B^d) between the pre-image of p under the hash function and the subset S is greater than $1/2$. In other words, the hash function does

not map too many points from the subset S to the same hash value p with a probability greater than 50%.

Here, the intersection $(h^{-1}(p) \cap S)$ represents collisions. It refers to the set of points in the domain S that are mapped to the same hash value p. The theorem says that the number of these collisions is limited and controlled, making the hash function suitable for various applications where a balance between maintaining relative distances in higher dimensions and minimizing collisions is desired.

The proof of Theorem 3 can be obtained using Lemma 5, Lemma 6, Lemma 7 and Lemma 8.

Lemma 5 Each $h \in \mathcal{H}_d$ is $\sqrt{d}$-expansive.

Proof: The proof for Lemma 5 follows the same approach as in the 2-dimensional case, but instead of using the d_1 distance metric, the d_2 distance metric (Euclidean distance) is employed. As a result, the expansiveness is scaled by a factor of $\sqrt{d}$ for higher-dimensional spaces.

Implication: The implication of Lemma 5 is that in higher-dimensional spaces, each hash function in the family $\mathcal{H}_d$ can separate points that are close to each other by at least a factor of $\sqrt{d}$. This property is useful for locality-sensitive hashing in higher-dimensional spaces, as it ensures that points that are closer together in the original space will remain relatively close in the hashed space.

By proving that each hash function in $\mathcal{H}_d$ is $\sqrt{d}$-expansive, it is demonstrated that the hash family is well-suited for applications in which proximity preservation is crucial, such as nearest neighbor search, clustering, and other machine learning tasks.

Fact 2 Let $\mathcal{B}_d(r)$ be the d-dimensional ball of radius r centered at the origin, and let $\mathcal{S}_d(r)$ be the bounding sphere of $\boldsymbol{B}_d(\boldsymbol{r})$. Then, letting $\Gamma(x)$ denote a gamma function,

- the volume $|B_d(r)|$ of $\mathcal{B}_d(r)$ is equal to $\frac{2r^d}{d}\frac{\pi^{d/2}}{\Gamma(d/2)}$,
- the area $|\mathcal{S}_d(r)|$ of $\mathcal{S}_d(r)$ is $\frac{2r^{d-1}\pi^{d/2}}{\Gamma(d/2)}$.

Lemma 6 There exists a constant B_1 such that for any $p \in D$ and any $h \in \mathcal{H}_d$, there are at most B_1^d choices of $q \in D$ such that $d_2(p,q) < \sqrt{d}R$ and $h(p) = h(q)$.

Proof: After applying the rotation matrix but before the rounding off, consider:

1. We take a unit cube lattice about a given point. We want to represent the points that can be mapped to the same bucket because they are nearby.

2. The cubical lattice is bounded by a unit cube. Initially, we say that number of q for which h(p)=h(q) is bounded by the number of points intersecting the cube.
3. The maximum distance between two points within the lattice is $d^{1/2}$. Therefore, many points will lie outside the bounding cube which might be mapped to the same bucket.
4. In order to get a better bound we circumscribe the lattice with a ball of radius $d^{1/2}$. The points intersecting this ball are said to be likely to be mapped to the same bucket. This ball is drawn with respect to any one point in the lattice.
5. But if we consider a diagonal point to that point, it will be on the surface of the bounding ball. However, if that point can be mapped to the same bucket, a point $d^{1/2}$ distance further away from this point can also be mapped to the same bucket.
6. In order to overcome this, we circumscribe the lattice with a ball of radius $2d^{1/2}$. Now, we have a clear upper bound on the number of possible points q that can be mapped to the same bucket as the point p.
7. As we have a unit lattice, We can simply take the volume of the ball in order to get the bound (by using Fact 2 with $r = 2\sqrt{d}$):

$$\frac{2(2\sqrt{d})^d}{d}\frac{\pi^{d/2}}{\Gamma(d/2)} \leq \pi^{d/2}\frac{(2\sqrt{d})^d}{(d/2-2)!} \leq B_1^d.$$

   Now, we can round off the denominator in the second term to (d/2)! which can written as a power of d and hence will give us the bound B_1^d.
8. For the first part, we need to bound the number of points from d-dimensional unit lattice contained in any (possibly rotated) unit cuboid $\mathcal{C}$. For the second part, notice that each g splits the domain into disjoint cuboids of side at least $\Omega(R)$ such that no two points from the same cuboid overlap. Hence it is sufficient to bound the number of such cuboids intersecting a d-dimensional ball of radius $\sqrt{d}R$.

Implication:

The implication of Lemma 6 is that, for any point p and any hash function h in the higher-dimensional hash family $\mathcal{H}_d$, the number of points q that collide with p (i.e., have the same hash value) and are within a Euclidean distance (d_2 metric) of $\sqrt{d}R$ is bounded by a constant B_1^d. This result is significant for a few reasons:

- **Collision probability control:** Lemma 6 provides an upper bound on the collision probability for points that are close to each other in the original space. This property is important for locality-sensitive hashing since it helps ensure that the hash functions in the family can effectively distinguish between close points.
- **Bucket size:** By bounding the number of points q that can collide with p, Lemma 6 implies that the bucket size, or the number of points that can be mapped to the same hash value, is limited. This is crucial for efficient nearest-neighbor search and other applications, as it prevents the hash family from having too many false positives (i.e., points that are not actually similar but are placed in the same bucket due to the hashing function.
- **Performance in high-dimensional spaces:** Lemma 6 shows that the hash family's performance scales with the dimensionality of the space, as the bound on the number of colliding points grows with the dimension d. This property is essential when working with high-dimensional data, as it ensures that the hash family remains effective even in more complex spaces.

In summary, Lemma 6 demonstrates that the hash family $\mathcal{H}_d$ has desirable properties for locality-sensitive hashing in higher-dimensional spaces, such as controlled collision probability, bounded bucket size, and scalable performance with dimensionality.

Lemma 7 Let $p, q \in D$ be such that $d_2(p,q) \geq \sqrt{d}R$. Then, there exists a constant A such that

$\Pr(r(p), r(g)$ are equal in k coordinates $) \leq \sqrt{d}\left(\begin{array}{c} d \\ k \end{array}\right)\left(\frac{A}{R}\right)^k.$

Implication: The implication of Lemma 7 is that it provides an upper bound on the probability that two points p and q in D, which are at least $\sqrt{d}R$ apart in Euclidean distance, will collide in k coordinates after applying the rotation function r and rounding-off. The bound is given by $\sqrt{d}\binom{d}{k}\left(\frac{A}{R}\right)^k$, where A is a constant.

This lemma is significant for a few reasons:

- **Distinguishing far-apart points:** Lemma 7 demonstrates that when two points are far apart (at least $\sqrt{d}R$ away from each other), the bound which represents an upper bound on the collision probability(not the actual probability) will increase. However, the actual probability that they collide in k coordinates decreases as their distance increases. This property is crucial for locality-sensitive hashing, as it helps ensure that the hash functions in the family can effectively differentiate between points that are far apart.

- **Performance in high-dimensional spaces:** Lemma 7 shows that the hash family's performance scales with the dimensionality of the space, as the bound on the collision probability grows with the dimension d. This property is essential when working with high-dimensional data, as it ensures that the hash family remains effective even in more complex spaces.
- **Balancing false positives and false negatives:** By providing an upper bound on the collision probability for points that are far apart, Lemma 7 helps strike a balance between false positives (colliding distant points) and false negatives (non-colliding close points). This balance is crucial for various applications, such as nearest-neighbor search, where minimizing both false positives and false negatives is important for efficient and accurate results.

Lemma 8 There exists a constant C_2 such that for any $p, q \in D$, with $d_2(p, q) \geq \sqrt{d}R$

$$\Pr(h(p) = h(q)) \leq \left(\frac{C_2}{R}\right)^d$$

Implications: The implications of Lemma 8 are as follows:

- **Separation of far-apart points**: Lemma 8 ensures that the probability of hashing two far-apart points (with a distance of at least $\sqrt{d}R$) to the same hash value is small. This property is crucial for locality-sensitive hashing, as it enables the hash family to separate points that are far apart effectively, thus reducing false positives in applications such as nearest-neighbor search.
- **Scaling with dimensionality:** The upper bound on the collision probability in Lemma 8 depends on the dimensionality d of the data points. This result indicates that the hash family's performance scales with the dimensionality of the space, making it suitable for high-dimensional data.

4 Negative Results

In this section, we will highlight the limitations of non-expansive hash functions as presented by the authors of the paper. They discuss the impact of these limitations on achieving small bucket sizes and focus on two important aspects: Lower Bounds on Collision Probability and Lower Bounds on Multifoldings. The concept of multifoldings refers to the composition of foldings and translations in hash functions, which can be considered a subclass of non-expansive functions.

Lower Bounds on Collision Probability:

The authors present key results such as Theorems 5, 6, and 7, which outline that for any family of non-expansive functions, there exists a pair of elements with a significant collision probability. This demonstrates the inherent limitations of non-expansive functions in achieving small bucket sizes, regardless of the distance metric used.

Lower Bounds on Multifoldings:

The authors also discuss the impact of multifoldings on bucket sizes. They present Theorem 8, which shows that even when hashing small sets, hash function families consisting of simple multifoldings have large bucket sizes. This further emphasizes the challenge of achieving small bucket sizes with non-expansive hash functions.

5 Summary and Open Problems

The paper investigates the possibility of achieving small bucket sizes using non-expansive hash functions. The authors present lower bounds that suggest that small bucket sizes cannot be achieved with non-expansive functions, particularly for a subclass of non-expansive functions called multifoldings. They provide a series of theorems and lemmas to support their argument.

Also, the authors present a locality-preserving hashing scheme with set size $\Theta\left(R^{d/2}\right)$ and establish a bound of $O\left(R^{1-\epsilon}\right)$ for non-expansive hashing schemes based on multifoldings. They also provide a non-expansive hash family with collision probability $1/R$ (for d_∞) and demonstrate a matching lower bound for collision probability in non-expansive hashing for d_∞.

Open Problems and Further Directions:

- It remains an open problem as to whether any non-expansive function can be viewed as a multifolding. The authors assume that all the non expansive functions can be viewed as a multifolding and have given the lower bound on multifolding only.
- The results presented in the paper are focused on the case when $d = 2$, with a note that the results can be similarly handled for $d \geq 2$. Further exploration of cases where $d > 2$ may be needed to better understand the behavior of non-expansive functions in higher dimensions.

- Can a small bucket size with the set S size of $O(R)$ elements (rather than $O\left(R^{1/2}\right))$) be achieved with constant multiplicative and $\sqrt{d}$ additive expansion terms, if the dilation is allowed to be multiplicative?
  The hashing scheme should maintain a constant multiplicative expansion term (independent of dataset size or dimensions) and a $\sqrt{d}$ additive expansion term. The challenge is to achieve these characteristics when dilation is multiplicative rather than additive.
- The paper demonstrates that non-expansive hash functions have large bucket sizes even when hashing small sets. However, it would be interesting to explore the trade-offs between bucket sizes and other properties of hash functions, such as computation time and space requirements.
- Can the proposed hashing scheme be adapted to handle dynamic datasets, where elements are frequently added or removed, while maintaining locality preservation and performance guarantees?

References

[1] N. Linial and O. Sasson, Non-Expansive Hashing, In Proc. 28th STOC (1996), pp. 509-517.

[2] D. Knuth, The Art of Computer Programming, vol. 3 , Sorting and Searching, Addison Wesley, 1973

[3] T.M. Cover and P.E. Hart, "Nearest neighbor pattern classification," IEEE Transactions on Information Theory, 13 (1967), pp. 21-27.

[4] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, Wiley, 1973.

[5] Panel on Discriminant Analysis and Clustering, National Research Council, Discriminant Analysis and Clustering, National Academy Press, 1988.

[6] L. Devroye and T.J. Wagner, "Nearest neighbor methods in discrimination," Handbook of Statistics, vol. 2, P.R. Krishnaiah, L.N. Kanal, eds., North-Holland, 1982.

[7] S. Cost and S. Salzberg, "A weighted nearest neighbor algorithm for learning with symbolic features," Machine Learning, 10 (1993), pp. 57-67.

[8] A. Gersho and R.M. Gray, Vector Quantization and Signal Compression, Kluwer Academic, 1991.

[9] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," First International Conference on Knowledge Discovery and Data Mining, 1995.

[10] V. Koivune and S. Kassam, "Nearest neighbor fitters for multivariate data," IEEE Workshop on Nonlinear Signal and Image Processing, 1995.

[11] C. Buckley, A. Singhal, M. Mitra, and G. Salton, New Retrieval Approaches Using SMART: TREC 4. Proc. Fourth Text Retrieval Conference, National Institute of Standards and Technology, 1995.

[12] G. Salton, Automatic Text Processing, Addison-Wesley, Reading, MA, 1989.

[13] H. Hotelling, "Analysis of a complex of statistical variables into principal components", Journal of Educational Psychology, 27 (1933), pp. 417-441.

[14] S. Deerwester, S. T. Dumais, T.K. Landauer, G.W. Furnas, and R.A. Harshman, "Indexing by latent semantic analysis," Journal of the Society for Information Science, 41(1990), pp. 391-407.

[15] K. Karhunen. Über lineare Methoden in der Wahrscheinlichkeitsrechnung. Ann. Acad. Sci. Fennicae, Ser. A137, 1947.

[16] M. Loéve. Fonctions aleatoires de second ordre. Processus Stochastiques et mouvement Brownian. Hermann, Paris, 1948.

[17] C. Faloutsos, R. Barber, M. Flickner, W. Niblack, D. Petkovic and W. Equitz, "Efficient and effective querying by image content", Journal of Intelligent Information Systems, 3 (1994), pp. 231-262.

[18] A. Pentland, R.W. Picard, and S. Sclaroff, "Photobook: tools for content-based manipulation of image databases", In Proc. SPIE Conference on Storage and Retrieval of Image and Video Databases II, 2185, 1994.

[19] D. Dobkin and R. Lipton, "Multidimensional search problems," SIAM J. Computing, 5 (1976), pp. 181-186.

[20] K. Clarkson, "A randomized algorithm for closest-point queries," SIAM J. Computing, 17 (1988), pp. 830-847.

[21] S. Meiser, "Point location in arrangements of hyperplanes," Information and Computation (1993), 106, 2 , pp. 286-303.

[22] J. Kleinberg, "Two algorithms for nearest-neighbor search in high dimensions", these proceedings.

[23] H. Samet, The Design and Analysis of Spatial Data Structures, Addison-Wesley, Reading, MA, 1989.

[24] F.K. Agarwal and J. Matousek, "Ray shooting and parametric search," Proc. 24th STOC (1992), 517 - 526 .

[25] S. Arya, D. M. Mount, N.S. Netanyahu, R. Silverman, A. Wu, "An optimal algorithm for approximate nearest neighbor searching," Proc. 5th SODA (1994), pp. 573 582.

[26] J. Matousek, "Reporting points in halfspaces," Proc. 32nd FOCS, 1991.

[27] K. Mulmuley, "Randomized multi-dimensional search trees: further results in dynamic sampling," Proc. 32nd FOCS, 1991

[28] A.C. Yao and F.F. Yao, "A general approach to d-dimensional geometric queries," Proc. 17th STOC (1985), pp. 163 – 168.