A Study on Optimization of Similarity Search by R-tree using Dimension Reduction

Dewi Noorsyitta Azida Binti Abd Aziz[1,a)] Naoya Higuchi[1] Takeshi Shinohara[1]

Abstract: Similarity search is necessary in searching multimedia data such as images and sound. Moreover, hierarchical spatial indexes such as R-tree enable high-speed similarity search of large-scale multimedia data. Dimension reduction, which in this paper we are using Simple-Map (S-Map for short), improves the performance of R-tree by relaxing the curse of dimensionality. In this paper, we propose a method which can further improve the searching performance. First, we investigate the average and the standard deviation of the shrinkage ratio of the dimension reduction technique used. Then, by referring to those data, we propose a method which uses expanded projection distances to an optimal level. We report the performance of proposed method by experiments. Experiments show that our proposed approach are able to improve the searching performance, however, the precision is slightly degraded.

Keywords: Similarity Search of Multimedia Data, Shrinkage Ratio, Expanded Projection Distance

1. Introduction

Nowadays, computers can be found not only in working areas such as office and schools, but this high-technology device has becoming dispensable in every household and even for every human being. Since the second half of the 20th century until recently, the performance of a computer including its computing power and its storage capacity has been dramatically improved. As a result of the improvement, it can now handle varieties of multimedia data such as documents, digital images and audio clips in large quantities. In current world where varieties of information are mixed, information retrieval technology has become necessary in order to find the information required by users from among the vast amount of information. For example instead of going to libraries, people nowadays tend to search things they want on digital library via web search engine such as Google and Yahoo. In future time, information retrieval technology, especially the multimedia information retrieval will be crucial and highly demanded as people will likely search everything through the internet.

Information retrieval techniques are important tools for obtaining data that users want to access. When searching for multimedia data, rather than using exact match searching, which is finding only the information that exactly match the queries, it is more important to use *similarity search*, which is finding for information that is similar. This is because, unlike human who can judge if a pair of data is the same only by looking at them, computers are unable to do that as in many cases computers judge a pair of exactly same data as a total different pair. In similarity search in metric spaces, objects within smaller distance are considered similar. Thus, similarity search is a task to find objects near to a given query object with respect to a given distance.

When the dimensionality of objects is m, the computational cost to measure distance between two objects is $O(m)$, and when the number of database objects is n, a naive similarity search by sequential manner costs $O(mn)$, which is unrealistic for larger m and n. Hierarchical index structures such as R-tree [1] and M-tree [2], [3] have been developed in order to weaken the effect of n. On the other hand, the *dimension reduction* technique is a method to avoid influence of m. The dimension reduction not only reduces distance computation cost but also relaxes “the curse of dimensionality”. For example, it is known that the efficiency of R-tree is decreasing when the dimension is increasing, but the performance can be improved if R-tree is constructed on projected objects into lower dimensional space by dimension reduction. Dimension reductions for Euclidean spaces include KL transformation or principle component analysis (PCA) [4] and FastMap [5]. On the other hand, dimension reductions such as HMap [6] and Simple-Map (S-Map for short) [7] are applicable to any metric spaces metricized by L_1 distance, Hamming distance, string edit distance and so on [8].

One of the most important properties of dimension reduction is that the distance between any two objects is not expanded after projection. In other word, in dimension reduction, the object is projected to a low dimensional data so that the projected distance does not extend with respect to the distance in original space (actual distance). Although the projected objects of low dimensions cannot completely maintain the actual distance relationship, it is important to reduce the information loss. As the projection distance does not extend the actual distance, it is guaranteed that distant objects in the projection space are far from the original space. Thus, this establishes the safety of pruning strategy that excludes any data, which is far away from a query in the projection space, from the search target without examining the actual

[1] Department of Artificial Intelligence, Kyushu Institute of Technology, Iizuka, Fukuoka, 820-0067, Japan
[a)] q676017n@mail.kyutech.jp

distance. However, if the shrinkage of the distance is large, the object outside the searching range actually becomes closer in the projection space, resulting in deterioration in searching efficiency. For PCA, analytically optimal projection can be obtained. However, for H-Map and S-Map, it has been known no analytically optimal solution, therefore, it is necessary to use random selection with evaluation function as a clue. Additionally, in this paper, we propose a method to reduce the effect of the shrinkage of distance by using expanded projection distance.

In this paper, we use S-Map as the dimension reduction. In S-Map, the reference object is selected as a pivot [9], [10], [11], and the distance between each object and the pivot is set as a coordinate value in projection space. Thereby, the number of coordinates is given as the number of pivots. Then, the number of pivots at this time is the dimensionality of the projection space and the distance between objects in the projection space is given as the L_∞ distance. When using S-Map, it is known that the distance in the projection space is shrunk to a certain level compared to its actual distance. Therefore, we first investigate the shrinkage ratio of the data used, and then using the data, we find its average, the standard deviation and its distribution graph. From those data, each object is shrunk to approximately half of its distance in average. Thus, we can say that it may be safe to expand the projection distance value to an optimal level. However, the proposed method might be slightly dangerous as not all objects have the same shrinkage ratio. Although a higher expands value can reduce the distance computation and searching time, precision may drop. Thus, it is necessary to find an optimal value to expand the projection distance.

2. Similarity Search

2.1 Metric Space

Similarity search is a searching process where to retrieve data from database that is similar to query. This process is realized by defining dissimilarity measure between data, called *distance*, and extracting data in order of distances from query points. The similarity search system is a system that judges the object which is near (small distance) to query to be similar, and it then extracts them from the feature space.

Let $\mathcal{D} = \mathcal{R}^n$ be the domain of objects, which is the entire feature space targeted by the similarity search database. Here, $\mathcal{R}$ and n is denoted as the whole real number and the number of dimension of data respectively. Let $\mathcal{M} = (\mathcal{D}, d)$ be a *metric space* for $\mathcal{D}$ and $d : \mathcal{D} \times \mathcal{D} \to \mathcal{R}$ be a total *distance function* indicating an index of dissimilarity measure between any two objects. In the similarity search we use in this paper, the function d satisfies the following *axioms of distance*, where $\forall x, \forall y, \forall z \in \mathcal{D}$:

(1) $d(x, y) \geq 0$ (Non-negativity)
(2) $d(x, y) = d(y, x)$ (Symmetry)
(3) $d(x, y) = 0 \iff x = y$ (Identity of Indiscernibles)
(4) $d(x, y) \leq d(x, z) + d(z, y)$ (Triangle Inequality)

The most important property of the above axioms is the triangle inequality.

Next, we explain on the distance measurement used in the similarity search. Let x be any object in the feature space. When $\mathcal{D} = \mathcal{R}^n$, x is represented by an n-tuple $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$. The following three distance measurement functions satisfy the axioms of distance:

$$L_1 \text{ distance} : D(x, y) = \sum_{i=1}^{n} |x_{(i)} - y_{(i)}|,$$

$$L_2 \text{ (Euclidean) distance} : D(x, y) = \sqrt{\sum_{i=1}^{n} (x_{(i)} - y_{(i)})^2},$$

$$L_\infty \text{ distance} : D(x, y) = \max_{i=1}^{n} |x_{(i)} - y_{(i)}|.$$

In this paper, the distance measurement in the original space (actual distance) is defined as the L_1 distance, and the distance measurement in the projection space is performed using L_∞ distance.

Additionally, a set of points equidistant from a certain is called an equidistant plane. Figure 1 below shows an equidistant plane from the origin in two-dimensional space. L_1 distance becomes a rhombus with sides parallel to the diagonal of the coordinate axis, while L_∞ distance becomes a square with sides parallel to the coordinate axis respectively.

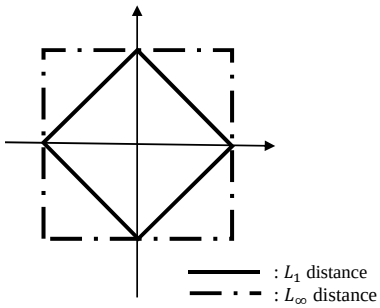

Fig. 1 Equidistant Plane

2.2 Similarity Query

A similarity query returns all objects which satisfy the selection conditions. In similarity queries, we seek objects that are close to the given query object. Suppose a collection $X \subseteq \mathcal{D}$. A *k-nearest neighbor*(*k-NN*) query returns the k closest elements to the query object q. The set R of results satisfies

$R \subseteq X, |R| = k$, and $d(q, x) \leq d(q, y)$

for any $x \in R$ and any $y \in X - R$. We can also define a *k-NN range* query. For a range r, it returns the k closest elements within a distance r. In the case of $k = 1$ for a *k-NN* range query, we call this an *NN range* query.

We now explain the *NN*-query for spatial indexes with a dimension reduction mapping. At first, we examine the distance between the query and object on the mapped space. When we find an object whose distance on the mapped space is smaller than the query range, we calculate the distance on the feature space. If the distance on the feature space is within the query range, we shrink the query range to the current distance and register the object as a temporary answer. Finally, after we found no objects within the

query range, we return the temporary answer as the optimal answer. At this point, we should note that filtering by distance in mapped spaces does not affect the accuracy of the search because $d'(x, y) \leq d(x, y)$ for any $x, y \in \mathcal{D}$, where d' is a distance function over mapped space.

3. Spatial Index Structure

3.1 B-tree

B-tree[12] is an index structure that realized an efficient search for one-dimensional feature space. It also supports dynamic operations such as insertion and deletion. Since it was proposed in 1970's, it has been widely used in the fields of databases and file systems. Generally, for m number of data, its searching performance is known to be $O(logm)$. In order to increase the efficiency of utilization of nodes in space, variants of B-tree such as B^*-tree and B^+-tree have been proposed.

3.2 R-tree

B-tree is an efficient index structure, but it can only be applied to one-dimensional feature space. However, with the progress of computer performance in recent years, the demand for similarity search on multidimensional feature space has increased. With that, many index structures that realize an efficient similarity search on multidimensional space have been invented. Among them, R-tree[1], proposed by Guttman, is one of the typical index structures and is based on the structure of B^+-tree.

R-tree divides the target space by an n dimension of hyper rectangle called MBR (Minimum Bounding Rectangle). n is the dimensionality of the feature. MBR is constructed so that objects that are spatially close to each other are stored in the same node as much as possible. An inner node of R-tree has information of MBR that stores all of its child nodes, and pointer to its child nodes. Meanwhile, a leaf node stores the pointer to an object in the database. Here, it is necessary to be aware that there is a possibility that MBRs, which have different nodes, might overlap each other [13], [14]. It is known that searching efficiency will deteriorate when constructing an R-tree for a high dimensional feature. In this way, as the dimension of the feature space increases, the shape of the MBR is separated from the normal shape, and the overlapping becomes large, effecting the searching efficiency to be deteriorated [15]. This problem is called *the curse of dimensionality*. In this paper, in order to calm the curse, we project a high dimensional feature space to a lower dimensional space by using the dimension reduction method S-Map.

3.3 Search using R-tree

For the searching process in R-tree, it visits only the nodes where the MBR and the query range intersect, in order of the distance between the MBR and the query point. Therefore, visits to unnecessary nodes can be prevented. However, in the case of constructing an R-tree using dimension reduction, in the leaf node, by checking the distance on the projection space before calculating the actual distance between object and query point, we can minimize the number of actual distance calculation times and more efficient search is realized.

Now we explain on the similarity search using R-tree. The searching process starts from the root node, and the nodes is searched using depth-first search. When the search is in a leaf node, first we calculate the projection distance to the object that is in the leaf node. If the distance is smaller than the search range, the search range value is updated to the distance value. When the search is in an inner node, the projection distance between the query and the MBR of its child node is calculated, and only the node with the MBR that intersects the search range is visited. At this time, the nodes that need to visit are inserted into the priority queue called Active Branch List (ABL). The nodes in the ABL are sorted in ascending order of the distance between the query and the MBR. By visiting the nodes from the beginning of the ABL, it is possible to visit from a node with a nearest MBR to the query. In this way, by controlling the visiting order of the nodes, it is able to accelerate the contraction of the search range of the query.

For example, in figure 2, we have MBR A, B, C and a query point. Consider the situation where all of the MBR A, B, C intersect the initial search range as shown in figure 2. When we are not considering the visit order, it visits everything in order of elements, in this example, it visits from A to B and then C. On the other hand, when controlling the visit order as described above, it visits MBR with the nearest distance from the query, which in this case is B, and contract the search range up to the distance between the query and the nearest object inside B. As a result, it does not have to visit MBR A. In this way, by controlling the visits order, the searching range can be contracted faster, and thus realizing an efficient search.

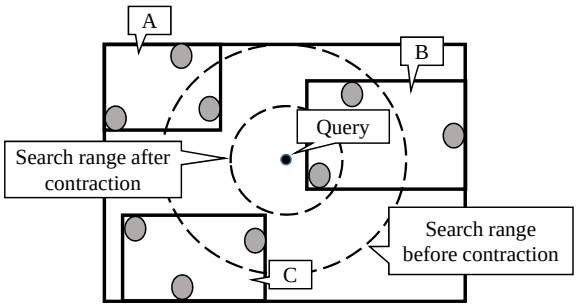

Fig. 2 Searching process by R-tree

4. Dimension Reduction

As described in Section 3.2, it is known that searching efficiency deteriorates when constructing an R-tree for a high-dimensional feature space. As the dimensionality of the feature space increases, searching efficiency becomes worse due to the curse of dimensionality. In order to relax this curse, *dimension reduction* techniques have been developed. By using dimension reduction, feature space is projected to a lower dimensional space while maintaining the distance between objects to some extent. Dimension reductions for Euclidean spaces include K-L transformation or principle component analysis (PCA) and FastMap. On the other hand, dimension reductions such as H-Map and S-Map are applicable to any metric spaces metricized by L_1 distance, Hamming distance, string edit distance and so on. In this paper, the space which the dimension is reduced by projection is called

projection space, while the space before the projection is called *original space*.

4.1 Projection Space

On the projection space, there is a possibility that the distance between any two objects shrinks compared to the distance in original space. We call this phenomenon as *shrinkage of distance*. For example, when an NN-query is executed in the projection space, there are cases where an object which its distance to the query point is shorter than the distance in real space will occur, and the object which is outside the search range in the original space is reduced to within the search range in the projection space when dimension reduction is applied. Figure 3 shows an example of distance shrinkage. The black circle represents an object. In figure 3, projection from two-dimensional space to one-dimensional space is performed using orthogonal projection onto coordinate axes. We can see that the distance on the one-dimensional space is shorter (shrinking) than the distance on the two-dimensional space. In this way, although the distance can be reduced, a function that may expand the distance is not suitable as a projection function. This is because, during the searching process, we only measure the distance in original space (called actual distance) when the objects are within the search range in the projection space. If the distance expands, it is impossible to filter such objects.

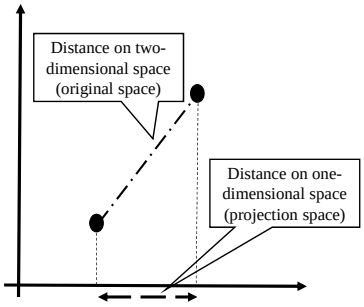

Fig. 3 Example of distance shrinkage

We denote projection space as $\mathcal{D}' = (\mathcal{S}', d')$. Here, $\mathcal{S}' = \{f(O_1), f(O_2), \ldots, f(O_i), \ldots, f(O_m)\} \subseteq \mathcal{D}$ and d' is a global distance function indicating the distance between any two objects on the projection space. $f : \mathcal{R}^n \to \mathcal{R}^{n'}$ is a projection function of object from the original space to the projection space. Here, $n'(< n)$ is the number of dimensions on the projection space. The following condition holds for the range query on the projection space due to the shrinkage of distance.

$$Range(\mathcal{D}', f(Q), r) = \{O_i \in S \,|\, d'(f(O_i), f(Q)) \leq r\}$$

$$Range(\mathcal{D}, Q, r) \subseteq Range(\mathcal{D}', f(Q), r)$$

In search of R-tree using projection, it visits only the nodes where the MBR intersects in the search range in projection space. When the search is in leaf node, the actual distance is only calculated only for objects that are within the search range in the projection space. Since the projection space has a relatively lower dimension than the original space, the cost of the distance function d' can be less than d. By using projection as described, it is possible to suppress the number of distance computation in original space, thus we can expect to improve the searching efficiency. However, in the projection space where shrinkage of distance occurs extremely, the number of times to compute the actual distance increases. Therefore, shrinkage of distance caused by dimension reduction is an important factor related to search efficiency. The merit is that it is applicable to all distance spaces that satisfy the axioms of distance.

4.2 Simple Map

The dimension reduction we use in this paper is S-Map. As described in 4.1, the projection function f is desirably a function that is not easily influenced by shrinkage of distance. S-Map uses the actual distance between one pivot and an object as a projection function, and projects it onto the L_∞ distance space.

We assume two metric spaces $(U, \mathcal{D})$ and $(U', \mathcal{D}')$, where $\mathcal{D}$ and $\mathcal{D}'$ are distance functions satisfying triangle inequality. Let $dim(x)$ for a data x denote the dimensionality of x. Then, we say that a mapping $\varphi : U \to U'$ is a *dimension reduction* if it satisfies the following conditions for every $x, y \in U$:

$$dim(\varphi(x)) < dim(x) \tag{1}$$

$$\mathcal{D}'(\varphi(x), \varphi(y)) \leq \mathcal{D}(x, y) \tag{2}$$

Condition (1) means that φ reduces the dimensionality of data, and condition (2) means that $\mathcal{D}'$ provides the lower bound of distance $\mathcal{D}(x, y)$, which guarantees to ignore a data without computing $\mathcal{D}$ in similarity search (safe pruning). For example, if $\mathcal{D}'(\varphi(q), \varphi(x))$ exceeds the current search diameter of a query q, then x can be ignored, or safely pruned, without computing $\mathcal{D}(q, x)$.

S-Map[7] is a kind of *Frechet embedding*, that any finite metric space of n points can be embedded isometrically into n-dimensional L_∞ normed space. A similar idea has also been proposed by Hjaltason and Samet, where for a point $p \in U$ called *pivot*, we define an S-Map φ_p of $x \in U$ with p as follows:

$$\varphi_p(x) = \mathcal{D}(p, x).$$

From the triangle inequality, the following inequality holds for every $x, y \in \mathcal{U}$:

$$|\varphi_p(x) - \varphi_p(y)| \leq \mathcal{D}(x, y).$$

Furthermore, using a set $P = p_1, \ldots, p_m$ of pivots, we define S-Map φ_p with P as follows:

$$\varphi_P(x) = (\varphi_{p_1}(x), \ldots, \varphi_{p_m}(x)).$$

Suppose that we give $\mathcal{D}'$ as follows:

$$\mathcal{D}'(\varphi_P(x), \varphi_P(y)) = \max_{i=1}^{m} |\varphi_{p_i}(x) - \varphi_{p_i}(y)|.$$

In other words, if the projected space U' is considered as an L_∞ metric space, and when m is smaller than the original dimension, then an S-Map φ_P becomes a dimension reduction.

In this paper, we apply S-Map to the feature quantity (64 dimensions) of the object extracted by the feature function, and the dimension is reduced to 8 dimensions. Then, the image of the object generated by S-Map is used to construct an R-tree.

5. Proposed Method

As described above, one of the most important properties of dimension reduction is that the distance between any two objects is not expanded after projection. Thus, by projecting objects with S-Map, the distance between those objects may shrink (and is not expanding too, from the triangle inequality) compared to original distance. This shrinkage, that is the distance deficiency, is desired to be small for similarity search. The shrinkage of the distance can be reduced by increasing the projection dimension. However, increasing the projection dimension will strongly influenced the curse of dimensionality, resulting in low efficiency similarity search. Thus, it is important to minimize the shrinkage of the distance in a lower dimension while maintaining a high efficiency of similarity search. The *distance preservation ratio* (DPR, for short) for a set S of pairs (x_i, y_i) of points is the following ratio of sums of distances.

$$\frac{\sum \mathcal{D}'(\phi(x_i), \phi(y_i))}{\sum \mathcal{D}(x_i, y_i)}.$$

While the *shrinkage ratio* for the set of points is as follows:

$$\frac{\mathcal{D}'(x_i, y_i)}{\mathcal{D}(x_i, y_i)}.$$

The DPR for the data that we use is approximately 0.54, where around 54% of the data distance is preserved in average. Moreover, we also investigate the shrinkage ratio of those data. Figure 4 shows the distribution graph of shrinkage ratio, where it is plot by actual and projection distance of randomly selected 500 samples.

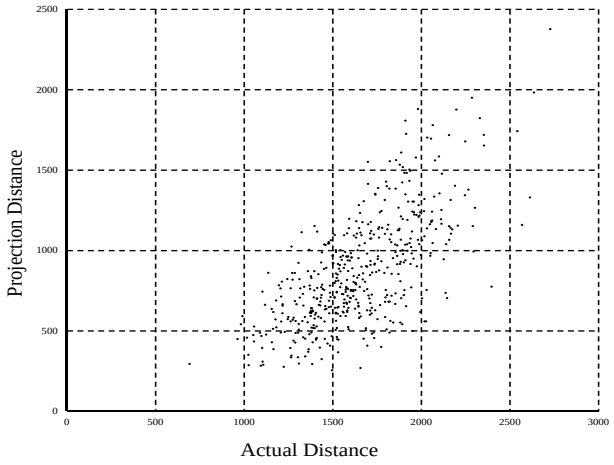

Fig. 4 Distribution of shrinkage ratio

In average, each object is shrunk to approximately 53% and the standard deviation σ is about 0.15. Thus, we can say that it is safe to expand the projection distance value up to twice the value. However, the proposed method might be slightly dangerous as not all objects have the same shrinkage ratio. The shrinkage ratio depends on the data and pivot used. Thus, there are some objects that shrink quite largely, and there are some that just slightly shrink. This can be proved from the maximum and minimum value of shrinkage ratio which is 0.97 and 0.12, respectively. Thus, a method to use expanded projection distance is proposed. The expand ratio of projected distance is given by

$$expand = \frac{1}{av + \alpha\sigma},$$

where av, σ and α are the average and standard deviation and multiplier for σ respectively.

There are mainly two places that we use expanded projection distance, during the search in inner node and leaf node. As described in section 3.3, in conventional similarity search for R-tree, when the search is in a leaf node, it calculates the projection distance to the object that is in the leaf node. If the distance is smaller than the search range, the search range value is updated to the distance value. Here, in conventional method, we compare the search range with projection distance. However, in this paper, we expand the value of projection distance by multiplying it with expand ratio and compare it with search range. Similarly, when the search is in an inner node, before insert the nodes that need to visit, we expand the projection distance and compare with search range. With this method, we can contract the search range faster and reduce the number of distance computation.

6. Experiments

In this section, we report experiments using images data, which is approximately 7 million 2D frequency spectrums of 64 dimension data extracted from about 2,900 videos. Randomly generated data are not appropriate for nearest neighbor experiments because in higher dimensional spaces, it is rare to find near data. Therefore, we prepare 3 types of queries: *near, middle, far* which are generated from randomly selected pairs from database with mixing noise ratio of 10%, 20%, 30%, respectively. For example, a near query q is a weighted sum of randomly selected data x and y from database with weight 10% and 90%, respectively. We prepare 100 queries for each noise level. The experimental result shows the average of these 300 queries.

Table 1 illustrates the computer environment used.

Table 1 The SPEC of the PC used in Experiment

CPU	Intel(R) Core(TM) i7-3770 3.40GHz
Memory	16GBytes

Table 2 shows the experimental results using the proposed method with 8 projection dimension of S-Map. α, Expnd shows the expand ratio of projected distance, Dist shows the expand value of project distance, Nodes shows the number of nodes visited, Dist shows the number of distance computation, Time shows the searching time in millisecond, and Prec shows the searching precision in %. The value in table 2 shows the average per query. The first line, which the expand ratio value is 1, in the table shows the results of conventional method.

Table 2 Experimental Result

α	Expnd	Nodes	Dist	Time	Prec
–	1	9352	351992	37	100
1.50	1.32	6318	114294	22	99
1.25	1.39	5741	92074	20	98
1.00	1.47	5178	73501	17	96
0.75	1.56	4658	57873	15	93
0.50	1.65	4211	46286	13	90

7. Conclusions

From the experiments, we confirmed that the run time is improved by using the proposed method, but, the precision is slightly degraded. As the expand ratio value increases, the number of distance computation reduced and the searching time becomes faster. However, the precision dropped. From Table 2, we can conclude that, for 8 projection dimension S-Map, expanding projection distance by 1.65 times is the optimal way, as we are able to search with 3 times higher speed while maintaining 90% of the precision as compared to conventional method.

In the experiments we reported, we only use images data as experimental data. Therefore, we should consider using other kind of data such as music and colors too. Additionally, Moreover, besides S-Map, we should also consider applying the propose method with other dimension reduction techniques, such as Sketch.

References

[1] A. Guttman: “R-trees: a Dynamic Index Structure for Spatial Searching”, in ACMSIGMOD International Conference on Management of Data. New York, NY, USA: ACM Press, pp. 47–57, 1984.

[2] P. Ciaccia, M. Patella, P. Zezula: “M-tree: An Efficient Access Method for Similarity Search in Metric Spaces”. Proc. VLDB’97, pp. 426–435, 1997.

[3] P. Zezula, P. Savino, G. Amato, F. Rabitti: “Approximate Similarity Retrieval with M-trees”. VLDB J. vol. 7, pp. 275–293, 1998.

[4] K. Fukunaga: “Statistical Pattern Recognition (Second edition)”. Academic Press, 1990.

[5] C. Faloutsos, Lin, K.I.:“A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets”. Proc. ACM SIGMOD’95 vol. 24, pp. 163–175, 1995.

[6] T. Shinohara, J. Chen, H. Ishizaka: “H-Map: A Dimension Reduction Mapping for Approximate Retrieval of Multi-dimensional Data”. Proc. DS’99, LNAI vol. 1721, pp. 299–305, 1999.

[7] T. Shinohara and H. Ishizaka: “On Dimension Reduction Mappings for Approximate Retrieval of Multi-dimensional data”, in Progress in Discovery Science, LNCS vol. 2281. London, UK: Springer, Verlag, pp. 224–231, 2002.

[8] M.M. Deza, E. Deza: “Encyclopedia of distances (Second edition)”. Springer, 2013.

[9] E. Chavez, G. Navarro, R. Baeza-Yates, J. Marroqu: “Searching in Metric Spaces”. ACM Comput. Surv. vol. 33, pp. 273?321, 2001.

[10] R. Mao, W. Miranker, P. D.:“Pivot Selection: Dimension Reduction for Distance-based Indexing”. J. Discret. Algo. vol. 13, pp. 32–46, 2012.

[11] R. Mao, P. Zhang, X. Li, X. Liu, M. Lu: “Pivot Selection for Metric-space Indexing, Internat”. J. Mach. Learn. Cybernet. vol. 7, pp. 311–323, 2016.

[12] R. Bayer and E. McCreight: “Organization and Maintenance of Large Ordered Indexes”. Acta Informatica, Vol. 1, pp. 173?-189, September 1972.

[13] K. Tashima, T. Aoki, T. Shinohara: “A Study on Data Allocate Method to Nodes for Acceleration of Similarity Search by R-tree”. The 62nd JCEEIE, 2009. (in Japanese)

[14] T. Aoki, K. Tashima, T. SHinohara: “A Study on Acceleration of Similarity Search by R-tree -Proposal of Data Allocation Method to Nodes-”. The 74th SIG-FPAI, 2009. (in Japanese)

[15] Michael Otterman: “Approximate matching with high dimensionality r-trees”. M. Sc. Scholarly paper, Dept. of Computer Science, Univ. of Maryland, College Park, MD, 1992.