Classification and Evaluation of High-dimensional Image Indexing Structures

Mohammadreza Keyvanpour, Najva Izadpanah, and Haleh Karbasforoushan

***Abstract*—In the recent years, lots of research on image databases has led to proposition of different kinds of multi-dimensional indexing structures to support similarity search in image database systems. Most of current commercial multi-dimensional image indexing structures cannot manage image data points in high dimensional spaces. Such that sequential scanning of these high-dimensional data points can be faster than using a multi-dimensional indexing structure. Therefore, another class of multi-dimensional indexing structures especially has developed for high-dimensional spaces. Due to importance and variety of high-dimensional indexing structures, we first classified them based on their main idea and then we evaluated them according to suggested design requirements of desired high-dimensional indexing structures. We hope this proposed framework will lead to development of more efficient structures to support similarity search in high-dimensional spaces.**

***Index Terms*—High-dimensional space, multi-dimensional indexing structure, image database.**

I. Introduction

Due to the advances in hardware technology and increase in the production of digital images in various applications, it is necessary to develop efficient techniques for storing and retrieval of such images. Content-based image retrieval is introduced to use the content of the image directly in the retrieval process. In this retrieval approach, visual features like color, shape and texture are extracted from each image object and mapped to multi-dimensional feature vectors. Therefore, images are represented as a vector of extracted visual features [1, 2]. A content-based image retrieval system performs image retrieval by computing similarities between images in image database and the image query, and returns the results, which are most similar to the image query. The similarity of two features is a function of their distance in the multi-dimensional space, and efficient similarity search is supported by a multi-dimensional indexing structure [1, 3].

Several multi-dimensional indexing structures have been proposed in the past decades. Increasing of the dimensionality strongly aggravates the quality of the multi-dimensional indexing structures. Usually these structures exhaust their possibilities until dimensions around 15[4]. However, it is shown that in high-dimensional data spaces, multi-dimensional indexing structures tend to perform worse than the sequential scanning of the data base. This fact is due to the curse of dimensionality phenomena [4], [5]. Hence new approaches have been recently proposed for indexing of multi-dimensional objects especially for high-dimensional spaces. Due to importance and variety of high-dimensional indexing structures, we classified and evaluated them in this study.

The rest of the paper is organized as follows: We describe multi-dimensional indexing structures in section 2. Then in section 3, the proposed framework for classification and evaluation of high-dimensional indexing structures is presented. And, Section 4 includes the conclusion.

Manuscript received April 17, revised May 20, 2012.

Mohammadreza keyvanpour is withDepartment of Computer Engineering, Alzahra University, Tehran, Iran (e-mail: Keyvanpour@alzahra.ac.ir).

Najva izadpanah izadpanah was with Department of Computer Engineering, Islamic Azad University,Qazvin branch, Qazvin, Iran(e-mail: n.izadpanah@qiau.ac.ir)

Haleh karbasforoushan was with Department of Computer Science, University of Southern California, Los Angeles, CA, USA.

II. Multi-dimensional Indexing Structures

Many multi-dimensional indexing structures have been proposed to improve the multi-dimensional image databases performance. Most of the multi-dimensional indexing structures are based on the principle of hierarchical partitioning of the data space, so that they have a tree-like structure [4]. In these structures, as shown in Fig. 1, data points are stored in data nodes and each directory node points to a set of sub trees. There is a single directory node, which is called the root. The index structures are height-balanced; it means the lengths of the paths between the root and all data nodes are identical [3], [6].

Based on how the data are partitioned in the space, multi-dimensional indexing structures can be classified into two categories: space partitioning-based indexing structures, and data partitioning-based indexing structures [7].

Space-partitioning-based indexing structures either partition the data space through assigning a hash code to each point (like what is done in Grid- Files), or hierarchically partition the data space in to sub regions by an iso-oriented hyper-plane, passing through a data point (like what is done in k-D-B-tree [3], [5], [6].

Data-partitioning-based indexing structures hierarchically partition the database into overlapping regions that enclosed either by a MBR (Minimum Bounding Rectangle) (like what is done in R-tree and R*-tree) or by a hyper-sphere (like what is done in SS-tree) [3], [6].

Unfortunately, overlap between regions degrades efficiency of data-partitioning-based indexing structures [3]. Some improvements are also proposed to reduce the overlap between regions; for example SR-tree uses intersection of hyper-rectangle and hyper-sphere instead of hyper rectangles to reduce overlaps between regions, and x-tree introduced super nodes to prevent overlaps between regions [3], [4].

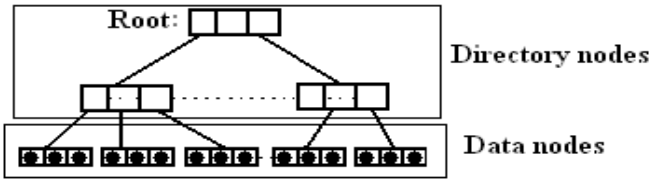

Fig. 1. Multi-dimensional indexing structure

Experience has shown that neither of these approaches is suitable for high-dimensional data space and sequential scan performs better than them in spaces with dimensions beyond 15[4].

III. Proposed Framework for Classification and Evaluation Of High-Dimensional Indexing Structures

This section includes the proposed framework for classification and evaluation of high-dimensional indexing structures. Components of this framework include both classification of high-dimensional indexing structures based on their main idea, and evaluation of these methods according to design requirements of desired high-dimensional indexing structures.

A. Classification of High-Dimensional Indexing Structures

According to our study on high-dimensional indexing structures, here, we have classified high-dimensional indexing structures based on their main idea. Our classification of high-dimensional indexing structures is shown in Fig. 2.

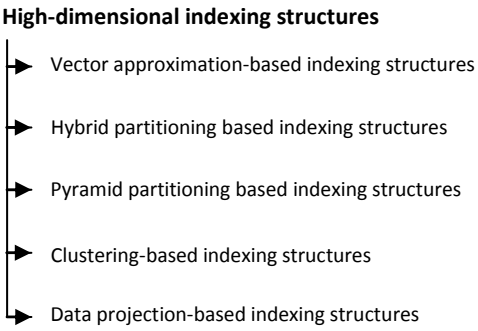

Fig. 2. Classification of high-dimensional indexing structures

1) Vector-approximation-based indexing structures

Due to the inefficiency of traditional multi-dimensional indexing structures in high-dimensional spaces, vector approximation is used to limit the size of the data. Vector approximation-based indexing structures use compression techniques like vector quantization in indexing of data points. The most common vector-approximation-based indexing structure is VA-File [8]. VA-File uses vector quantization to reduce the size of database and thus reduce the number of data nodes access during the search. An example of the VA-File structure with two dimensions is illustrated in Fig. 3. The data space is quantized into the cells and The VA-File stores the approximating value of every data point. Motivated by the inefficiency of VA-file for clustered databases, VA+-File [9] is proposed to use PCA transformation of data space that is more adapted to the clustered distribution of data points. In IQ-Tree [10], the idea of vector quantization was adopted to a hierarchical indexing structure. IQ-tree has a structure like R*-Tree, and in this structure data nodes are approximated using vector quantization. A-tree [11] is one of the most efficient vector approximation-based indexing structures. It also has a structure like R*-Tree, and in this structure MBRs of data nodes and data point are Approximated through proposed relative approximation. In relative approximation each MBR or data point is approximated by their relative positions in terms of parent's MBR. Representation of relative approximation of an MBR or a data point in the A-tree defines with the concept of VBR (Virtual Bounding Rectangle). In A-tree, the virtual bounding rectangle has been defined as in (1) [11].

$$V_i = (v, v')$$
$$v = (Q(b_1), Q(b_2), ..., Q(b_j), ..., Q(b_n)) \quad (1)$$
$$v' = (Q'(b_1), Q(b_2), ..., Q(b_j), ..., Q(b_n))$$

where, v is the vector quantization of the start point of the MBR_i relative to its parent's MBR, v' is the quantization of the start point of the MBR_i relatively to its parents MBR, Q and Q' are the quantization functions for start point and end point. b_j and b'_j $(j=1,..., n)$ are start point and end point of the MBR_i in n-dimensional space.

RA+-Blocks [12] is another improvement of VA-file on both uniform and clustered data sets. It divides data space into compact and disjoint regions and then uses RA-Blocks technique to approximate each region to improve VA-file on both uniform and clustered datasets.

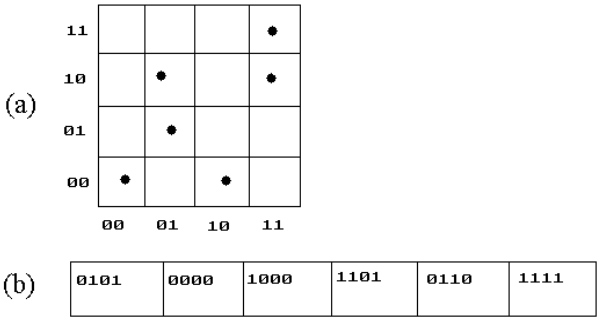

Fig. 3. An example of VA-File structure. (a): data space, (b): approximation points

2) Hybrid partitioning-based indexing structures

Hybrid partitioning-based indexing structures take advantages of both space-partitioning-based and data-partitioning-based indexing structures, as combines positive aspects of them in a single structure to face the problem of curse of dimensionality.

Hybrid tree [13] is proposed to use hybrid partitioning in order to achieve a scalable indexing structure in high-dimensional spaces. Hybrid tree recursively partitions data space into two overlapping sub spaces, using a single dimension. Hybrid tree performs insertion and deletion operations in the similar way to data-partitioning-based structures. Reference [14] has claimed that data points reside in the same node in Hybrid tree structure might not be spatially adjacent points in the real data space, thus it leads to increase in the number of disk access during search.

Motivated by this limitation, SH-tree [14] is proposed to include balanced nodes in hybrid tree structure to keep nearly spatially adjacent data points in the data space into each data node. An example of the SH-tree structure is shown in Fig. 4. SH-tree structure consists of three types of nodes: internal nodes, balanced nodes and data nodes.

3) Pyramid partitioning-based indexing structures

Pyramid partitioning-based structures use special pyramid partitioning to handle indexing of high-dimensional spaces. A pyramid technique [15] is proposed with the idea of transforming the multi-dimensional data points into one-dimensional values and then accesses the values using a one-dimensional indexing structure. As Fig. 5 is showing an example of pyramid partitioning in two-dimensional space, Pyramid technique first divides the data space into two-dimensional pyramids in such a way that the center point of the space are in their top, and in second step, cuts each pyramid into partitions parallel to the basis of it .Each partitions forms the data nodes of the indexing structure. Multi-dimensional data point as defined in (2), transforms by the pyramid that it belongs to and by its height in that pyramid [15].

$$PV(v) = i + h(v) \quad (2)$$

where, v is a multi-dimensional data point, PV (v) is pyramid value of a data point v, i is the index of the pyramid pi containing v and h (v) is the height of data point v within the pyramid Pi.

Unfortunately it has been shown in [15] that when the database has clustered distribution, the pyramid technique fails to provide efficient queries. To address this problem, the P+-Tree [16] is proposed. P+-tree is the generalization of the pyramid technique that divides the data base in different sub spaces and pyramid partitioning is performed on each of these sub spaces. Then at search time, query approximation will be performed. P+-tree has better performance for clustered database distribution.

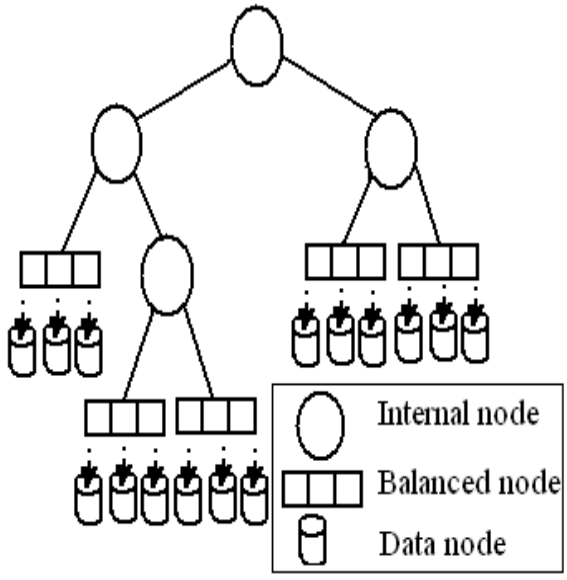

Fig. 4. An example of SH-tree structure

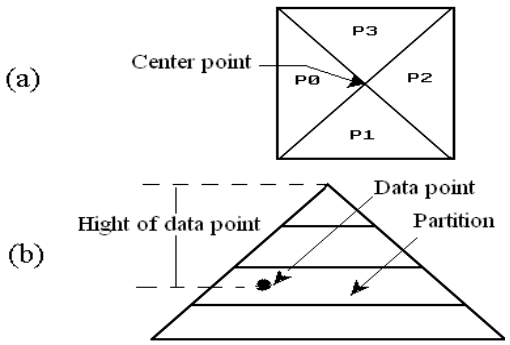

Fig. 5. Example of pyramid partitioning in two-dimensional space. (a): data space partitioning in to pyramids, (b): pyramid cut into partitions

4) Clustering-based indexing structures

The main idea of Clustering-based indexing structures first use clustering algorithms in order to cluster the data points, and then use approximation at search phase in such a way that search will be done in the obtained clusters which probably contain the nearest neighbors of the query point. The general architecture of the clustering-based structures is shown in Fig.6 Clustering-based indexing structures consist of two phases: clustering phase and search phase.

Clindex [7] is one of the examples of these methods that can achieve efficient approximate similarity search in high-dimensional spaces, by taking advantage of the clustering structures of a data set, and also sequential disk IOs such that store each cluster in a sequential file. At clustering phase, Clindex uses a grid-based clustering algorithm called CF (cluster-forming) for partition the data space. Then, obtained clusters are indexed through their centroid and indexing structure is built to support fast access to the clusters at search phase. Reference [17] has proposed a density-based clustering method that relies on the estimation of the distribution of the data points for indexing high-dimensional data bases. At clustering phase, it builds a statistical estimate of the density of the data to construct a clustered index. It uses a mixture of Gaussians to model data with estimation of the density. And at search phase, it uses a model to scan the clusters that probably contain the nearest neighbors of the query point. Reference [18] has proposed a new clustering-based indexing structure. At clustering phase, using hierarchical clustering algorithm PDDP, the data set is divided into two sub clusters. In such a way that, it projects all the data points on to first principal component, using (3), and divides the data into two sub clusters by the sign of projections [18].

$$F(x_i) = U^T (x_i - w) \quad (3)$$

where, $F(x_i)$ is projection function, x_i is multi-dimensional data point, U is first principal component, and W is centroid of the data.

After the algorithm has split the initial dataset into two sub clusters, it is going to recurse this procedure for one of the two sub clusters. This partitioning strategy creates a binary tree whose leaves are the final clusters of the data set. MBRs are the bounding forms which have been used to enclose each obtained cluster at each level of partitioning. In order to have no overlap between two cluster's MBR, they changed reference mark to have MBRs directed according to the first

principal component. At search phase, they proposed to use a k-nearest neighbors search adapted to the obtained hierarchical of clusters.

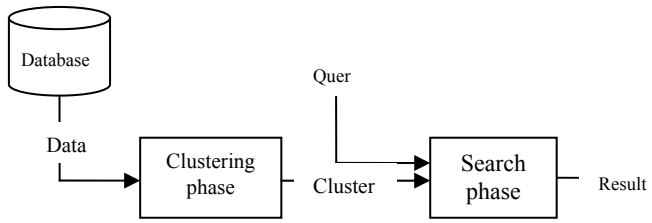

Fig. 6. General architecture of clustering-based structures

5) Data projection-based indexing structures

Main idea of data projection-based indexing structures is projecting dataset to several low dimensional spaces, and then run the query processing on all of those low dimensional spaces. Thus dimensionality cannot have negative effect to them.

MedRank [19] has proposed to use Random line Projection to project the database on to random lines. An example of projection into two random lines is shown in Fig. 7 After performing the projections, MedRank uses one-dimensional indexing structure B+-tree to index every random line. Then efficient k-nearest neighbor queries are performed and the data points nearest to the projected query point are returned.

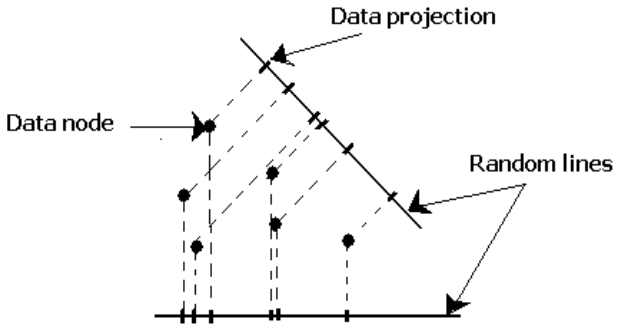

Fig. 7. An example of projection into two dimensions

Reference [20] has proposed to use locality sensitive hashing function to project the data in to the buckets. A hash function is locality sensitive if the probability of collision of two points is related to their similarity. Therefore, by using locality sensitive hash functions, similar points are hash to the same bucket .Then it has proposed to perform an approximate similarity search at query time in such a way that by finding the bucket that the query point hashes to that, all other points in that bucket will be returned as the result set.

B. Evaluation of high-dimensional indexing structures

According to our study on high-dimensional indexing structures, we present main design requirements of high-dimensional indexing structures as following**:**

Dynamicity: A desired high-dimensional indexing structure must have a dynamic structure. It means that structure incrementally constructed as the data points are inserted in the database.

Simplicity of the structure: A desired high-dimensional indexing structure must have a simple structure.

Query exactness: A desired high-dimensional indexing structure must use reasonable Query approximation to preserve high quality of results.

Support for clustered datasets: A desired high-dimensional indexing structure must support the datasets with clustered distribution. Because by increasing the dimensionality of datasets they tend to be more clustered, it is important for high-dimensional indexing structures to preserve their efficiency for clustered datasets.

High Selectivity: A desired high-dimensional indexing structure must support high selectivity.

Selectivity defines the average proportion of file pages that are actually pruned during the search. In fact, high-dimensional indexing structures must provide low number of file access during the search because accessing to file pages reduces the response time to a given query and increase the selectivity.

Low CPU time: A desired high-dimensional indexing structure must support low CPU time. CPU time defines the time that an indexing system takes for processing its operations in CPU and it is different with response time. Therefore, low CPU time guarantee efficient time complexity.

In this study, we analyzed high-dimensional indexing structures, and evaluated them according to above requirements.

Table I is showing our evaluation results.

TABLE I: EVALUATION OF HIGH-DIMENSIONAL INDEXING STRUCTURES

High-dimensional indexing structures	**Design requirements**					
	Dynamicity	***Simplicity***	***Query exactness***	***Support for clustered datasets***	***High Selectivity***	***Low CPU time***
Vector approximation-based indexing structures	good	medium	good	medium	medium	medium
Hybrid partitioning-based indexing structures	good	good	good	good	medium	medium
Pyramid partitioning-based indexing structures	good	poor	medium	medium	good	good
Clustering-based indexing structures	poor	medium	medium	good	good	poor
Data projection-based indexing structures	good	poor	poor	good	good	medium

IV. Conclusion

Similarity search in image databases requires special support at physical level. As dimensions of the data space grow, more considerations are required. A major challenge regarding this issue is employing some methods which facilitate data accessing and accelerate search operations in the database containing such high-dimensional data. So far, many multi-dimensional indexing methods for handling indexing of image data is developed. Considering in most applications, data spaces with high dimensions are used, and inefficiency of traditional multi-dimensional indexing structures in high-dimensional spaces, Recently a bunch of other structures, especially for high-dimensional spaces are developed. Given the importance and variety of these methods, in this study, a systematic framework for classify and evaluate high-dimensional indexing structures has suggested. According to our study on high-dimensional indexing structures, we classified them into five classes based on their main idea. Then we presented design requirements of high-dimensional indexing structures, and finally evaluated five classes of high-dimensional indexing structures based on suggested design requirements.

References

[1] R. DATTA, D.JOSHI, J. LI, and J. Z. WANG, "Image Retrieval: Ideas, Influences, and Trends of the New Age," *ACM Transactions on Computing Surveys*, volume 40,No. 2, pp. 1-60,2008.

[2] K. CHakrabarti, "Managing Large Multi-dimensional Datasets Inside a Database System," P.h.D dissertation , University of Illinois at Urbana-Champaign. Urbana, Illinois, 2001.

[3] C. Bohm, S. Berchtold, and D. A. KEIM, "Searching in High-Dimensional Spaces-Index Structure for Improving the performance of Multimedia Databases," *ACM Computing surveys*, vol.33, No.3, pp. 322-373,2001.

[4] K. Markov, K. Ivanova, I.Mitov, and S. Karastanev, "Advance of the Access Methods," *International Journal of Information Technologies and Knowledge*, Vol.2, 2008.

[5] N. Bouteldja, V. Gouet – Brunet, and M.Scholl, "Back to the Curse of Dimensionality with Local Image Descriptors", CEDRIC Research Report, 2006.

[6] V. Gaede and O. Günther, " Multi-dimensional Access Methods,"*ACM Computing Surveys*, Vol. 30, No. 2, 1998, doi:10.1145/280277.280279.

[7] C. Li, E. Chang, H. Garcia-Molina, and G. Wiederhold, " Clustering for Approximate Similarity Search in High-dimensional Spaces," *IEEE Transactions on Knowledge and Data Engineering*, 2002,pp. 792 – 808, doi:10.1109/TKDE.2002.1019214.

[8] R. Weber, H.-J. Schek, and S. Blott, " A quantitative analysis and performance study for similarity-search methods in high- dimensional spaces," *in Proc. International Conference on Very Large Data Bases (VLDB '98), CA, USA,* 1998, pp. 194–205.

[9] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. E. Abbadi, "Vector Approximation Based Indexing for Non-uniform High-dimensional Data sets," *in Proc. International conference on Information and knowledge management (CIKM '00)*, pp. 202–209, 2000.

[10] S. Berchtold, C. Bohm, H. V. Jagadish, and H.-P. Kriegel, and J. Sander, "Independent Quantization: An index Compression Technique for High-dimensional Dataspaces,"*in Proc. International Conference on Data Engineering(ICDE '00), USA*, pp. 577 - 588 , 2000.

[11] Y. Sakurai, M. Yoshikawa, S. Uemura, and H. Kojima, "The A-tree: An Index Structure for High-dimensional Spaces Using Relative Approximation,"*in Proc. International conference on Very Large DataBase (VLDB '00), Egypt*, 2000.

[12] I. Daoudi, S.E. Ouatik, A. El Kharraz, K. Idrissi, and D. Aboutajdine, "Vector Approximation Based Indexing for Hhigh-dimensional Multimedia Databases," Engineering Letters, 16:2, EL_16_2_05, 2008.

[13] K. Chakrabarti and S.Mehrotra, "The Hybrid tree: A Structure for High- dimensional Feature Spaces "*in Proc. IEEE international conference on data engineering,* pp. 440–447, 1999.

[14] D. T. Khanh, "The SH-Tree: A Novel and Flexible Super Hybrid Index Structure for Similarity Search on Multi-dimensional Data," *International Journal of Computer Science & Applications* , Vol. III, No. I, pp. 1 – 25, 2006.

[15] S. Berchtold, C. Bohm,and H.-P. Kriegel, "The Pyramid-technique: Towards Breaking the Curse of Dimensionality, "*in Proc. ACM SIGMOD International Conference on Management of data Volume 27 Issue 2,* June 1998.

[16] R. Zhang, B. C. Ooi, and K.-L. Tan. "Making the pyramid technique robust to query types and workloads," *in Proc.International conference on Data engineering,* pp.313-324,2004.

[17] K. P. Bennett, U. Fayyad,and D. Geiger, "Density-based indexing for approximate nearest neighbor queries,"*in Proc. ACM SIGKDD International conference On Knowledge discovery and data mining(KDD '99),* pp. 233–243, 1999.

[18] M. Taileb, S. Lamrous,and S. Touati, "Non-overlapping Hierarchical Index Structure for similarity search," *International Journal of Computer Science,* Vol. 3 ,No. 1, 2007.

[19] R. Fagin, R. Kumar,and D. Sivakumar, "Efficient Similarity Search and Classification via Rank Aggregation," *in Proc. ACM SIGMOD International Conference on Management of data(SIGMOD '03),* pp. 301–312, 2003.

[20] A. Gionis, P. Indyk,and R. Motwani, "Similarity search in high dimensions via hashing ,"*in Proc. International Conference on Very Large Data Bases(VLDB '99),* pp. 518–529, 1999.