Combining Metric Cache and D-cache-enhanced MAMs for Maximizing Efficiency of Similarity Search

K.Karthik, G.Pradeep Reddy
Mtech 2nd year , Lecturer
Dept of cse, jntuacea,Dept of cse, jntuacea

Abstract

Caching is used in computer applications to speed up the performance data retrieval. It is widely used in database management systems where queries are very frequent for reducing I/O cost. However, the situation is dramatically different in modern complex databases consisting of snapshots of nature (i.e., images, sounds, or other signals), like multimedia databases, bioinformatics databases, time series, etc. Here, people often adopt the similarity search within the content-based retrieval paradigm; (MAMs) are suitable in situations where the similarity measure is a metric distance. In particular, metric access methods (MAMs), used for similarity search in complex unstructured data, have been designed to minimize rather the number of distance computations than I/O cost while making indexing or querying.Recently Skopal, Lokoc, and Hustos proposed D-cache with MAMs such as Pivot tables, M-tree and D-file to improve performance of query processing. In this paper we propose a combined approach that uses metric cache in front of D-cache enhanced MAMs for boosting the performance of similarity search further. We built a prototype in Java platform to demonstrate the proof of concept. The empirical results reveal that the hybrid approach yields better performance.

Index Terms –MAMs, D-Cache, index-free search, metric cache, similarity search

INTRODUCTION

Efficient data retrieval plays an important role in modern database management systems. In order to reduce I/O operations and thereby eliminate efficiency issues, caching is used in databases. Both caching and indexing strategies [1], [2] are widely used along with storage layouts [3]. These are considered while designing database management systems. RDBMS has become a stable alternative for storing and retrieving valuable business data permanently. In such systems, caching is used to improve query performance. They also realize query performance with various indexing strategies in place. End users need speed in query processing. Delay in processing causes more user wait time that is not expected by end users. Users are too good to see the results as faster as possible. In this context disk caching is highly efficient where queries are very frequent with respect to user sessions. Disc caching has been used for reducing I/O cost incurred while making queries on relational databases. However, in the real world there might be scenarios to use databases containing multimedia content like sounds, images, and other information. For instance time series databases, bioinformatics databases and multimedia databases have already overcome the

problem of disk I/O and they focused on distance computations. In such databases similarity search is often used and their needs improvement in processing such queries. Indexing techniques have been used traditionally in order to speed up query processing. The strategies used for indexing are general or domain specific. Whatever be the nature of indexing, it is basically to improve the query processing. Distributed indexing strategies [4] are also developed for speeding up similarity queries made for distributed databases. In is a fact that the data retrieval performance is affected more by CPU cost than I/O cost. Therefore in case of similarity search queries it is important to reduce CPU cost rather than I/O cost [5], [6]. In this context, it is minor issue with I/O cost incurred while performing similarity searches. Here the computation cost is considered to be the number of computations need for answering a specific query. Metric Access Methods (MAMs) are suitable candidates for accessing data based on similarity searches that needs to use distance metrics. The properties of metrics can help to have database organized in memory in order to process queries faster [7], [3], [1], [2]. Later on indexing is also used to answer similarity queries faster. The queries might be k-nearest neighbors (kNN) or other queries like content based image retrieval queries which are modeled after Query by Example (QBE) concept. MAMs are extremely useful in processing such queries as they eliminate unnecessary equivalence classes from the process of search. Reduced computation of query is achieved with MAMs as explored in [8], [9], [5], and [6].

In [10] D-Cache is used along with enhanced MAMs such as D-file, M-tree, and Pivot table in order to improve query processing with respect to similarity searches. In this paper we enhance the system further by using metric cache along with D-cache. Our contributions are as given below.

1. We propose metric cache that is placed in front of D-cache for improving the similarity search on metric data assets.
2. We propose an architecture which acts as a framework for accommodating D-cache and metric cache for processing similarity searches.
3. We built a prototype application in Java platform that demonstrates the proof of concept. The application makes use of huge amount of unstructured data to test the efficiency of the proposed system.

The remainder of the paper is structured as follows. Section II reviews literature pertaining to caching, D-cache, MAMs and other relevant topics. Section III provides information related to the proposed system. Section IV presents experimental results while section V concludes the paper.

RELATED WORKS

MAMs have been around for some time. However, distance cache for general purpose MAMs is the first time experimented in [10]. In this paper we improve them further with combination of metric cache. There has been caching concept in computer applications which buffers the frequently used data so as to avoid retrieving from hard disk or databases. This will decrease I/O cost dramatically. Caching in the context of databases with various kinds of data is explored in [11] and [12]. Cached distances are used for range queries where M-tree is used as underlying data structure as proposed by Kailing et al. [13]. When the distances are computed from route object to the query, they are cached for future retrievals.

This will improve performance of such queries. As the performance gain is more the memory cost of this is tolerable [13]. However, caching of unnecessary distance measures is not done in order to optimize performance further. Batch indexing and querying was also used. Bulk loading f data is part of many applications where good performance is required. To deal with such data two algorithms were proposed in [14]. They made use of index structures for boosting search performance such as Slim-tree and R-tree. The drawback is that the performance is degraded once new objects are inserted into the tree. To overcome this drawback M-tree was proposed which can support parallel insertion of objects. In [15] another approach was proposed to improve efficiency of MAMs. Instead of issuing multiple queries a batch of queries can be used in order to reduce computational andI/O cost. In [10] the D-cache technique proposed reduced I/O cost and also CPU cost through MAMs which have been enhanced to work with D-cache. K-nearest neighbor graph concept is used in [16] forperforming kNN queries.

Query result caching plays an important role in speeding up similarity searches. This idea was explored in [17] and [18]. History of similarity queries and answers can be stored in metric cache for performance improvement. In [10] D-cache is used for performanceimprovement of similarity searches. They also hint about the usage of high level metric class in front of D-cache with enhanced MAMs. In this paper we implemented the combined cache where metric cache is in front of D-cache to enhance the query processing speed.

PROPOSED FRAMEWORK

Before presenting our new framework for processing similarity searches we would like to introduce D-cache, enhanced MAMs briefly. However, more information can be found in [10]. D-cache is a data structure which resides in main memory which is meant for storing the distances computed by MAMs. The D-cache stores distance computations in the form of triplets as shown below where the first two are unique identifiers for objects while the third one is the distance between the objects.

$$[id(o_i), id(o_j), \delta(o_i, o_j)],$$

The content of D-cache is made up of many such triplets. However, it can also be visualized as a sparse matrix as given below where rows and columns represent objects and the cells and their values indicate the distances between objects.

$$D = \begin{array}{c} \\ o_1 \\ o_2 \\ o_3 \\ \dots \\ o_m \end{array} \begin{array}{c} \begin{array}{ccccc} o_1 & o_2 & o_3 & \dots & o_n \end{array} \\ \begin{pmatrix} & \delta_{12} & \delta_{13} & \dots & o_n \\ \delta_{21} & & & \dots & \delta_{2n} \\ & & & & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \delta_{m1} & & \delta_{m3} & \dots & \end{pmatrix} \end{array}$$

The operations performed on D-cache include distance retrieval and distance insertion. The enhanced MAMs used by D-cache include D-file which is best used for range query and kNN query, pivot tables and M-tree. More details and algorithms about the usage of D-cache with enhanced MAMs can be found in [10].The proposed framework combines both Metric Cache and D-Cache in order to improve performance of range queries and kNN queries. The architectural overview of the proposed approach is as shown in figure 1.

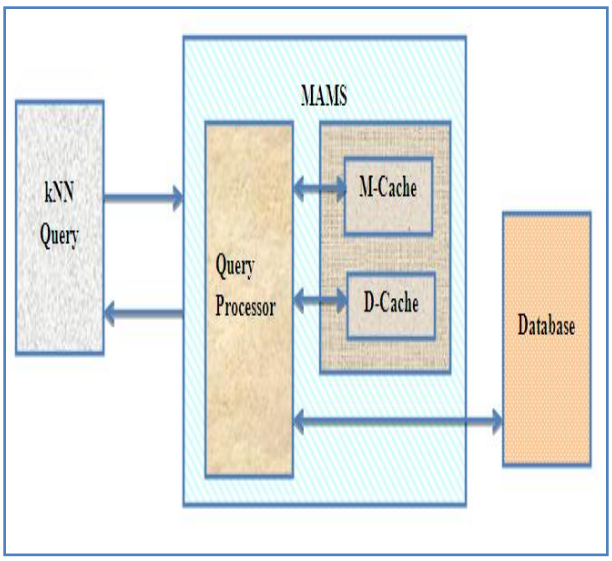
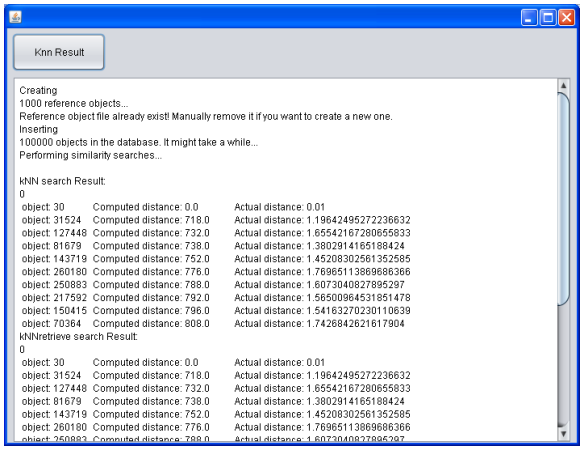


Fig. 1 –Proposed architecture that combines D-Cache and metric cache

As can be seen in figure 1, it is evident that two kinds of caches are in place for better performance. The metric cache is basically for storing history of similarity queries. The details stored in metric cache include descriptors and ids of database objects that have been returned by queries. When a new query is encountered, the metric cache returns result if exact value is found in metric cache. The metric cache might give a single value or multiple values based on the queryrequirements. Sometimes it may giveapproximate answer by combining a near correct value with other values. In such cases large retrieval error is likely to occur and the metric cache forwards that query to the retrieval system while updating itself. The D-cache on the other hand stores database objects, and their distances. They play a role to work with enhanced MAMs.

PROTOTYPE IMPLEMENTATION

We built a prototype application in Java platform with graphical user interface. The environment used is a PC with 4GB RAM, core 2 dual processor running Window



s 7 operating system. The result of kNN query is as shown in figure 2.

Fig. 2 – Result of kNN query

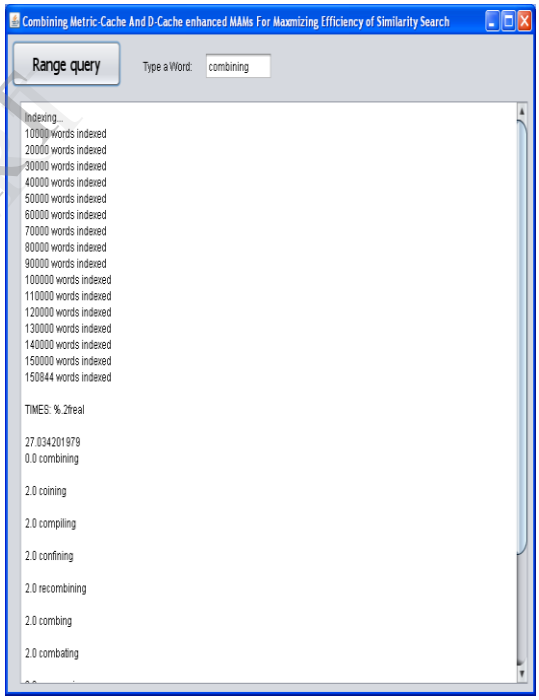


Fig. 3 – Result of range query

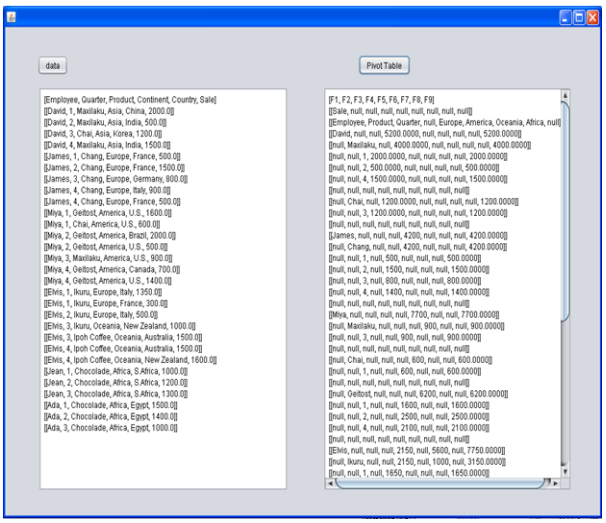


Fig. 4 – Data and corresponding pivot table

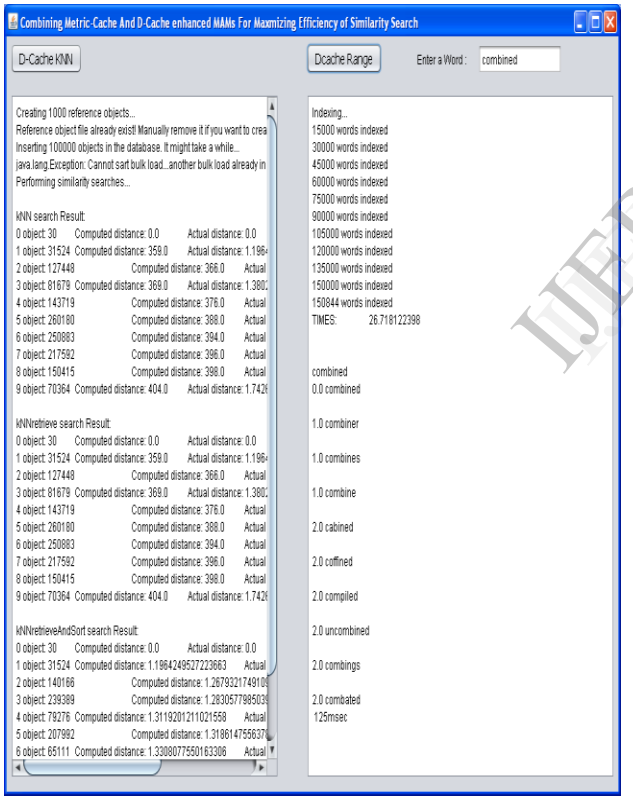


Fig. 5 –Result of kNN and range queries that use D-cache and metric cache

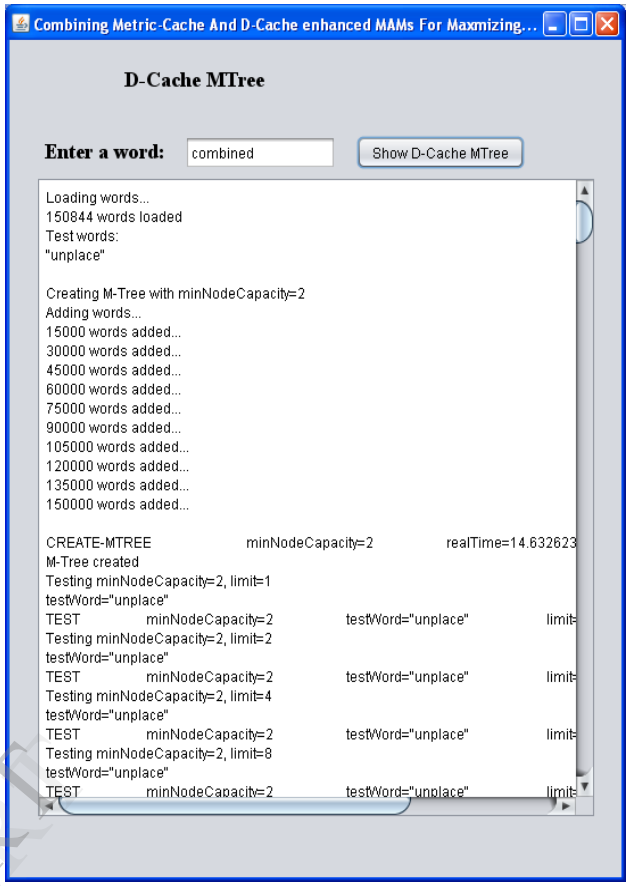


Fig. 6 – D –Cache and metric cache with M-tree MAM

EXPERIMENTAL RESUTLS

We made experiments without prototype application. The application demonstrates kNN and range queries with enhanced MAMs with D-cache. It also supports the same queries with the proposed combination of metric cache and D-cache. The experimental results are presented in the following graphs.

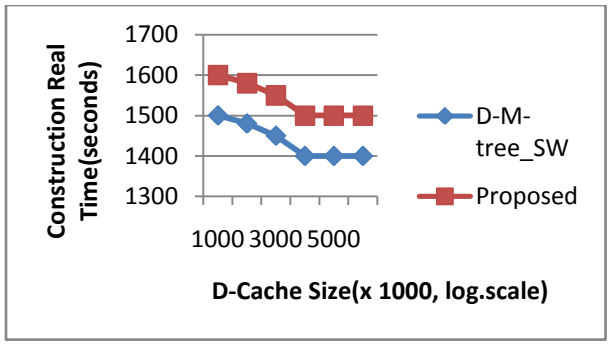


Fig. 2 – Construction of M-tree using single-way leaf selection

As shown in figure 2, horizontal axis represents D-Cache size while the vertical axis represents construction real time.The results reveal that the proposed shows higher performance than the existing approach.

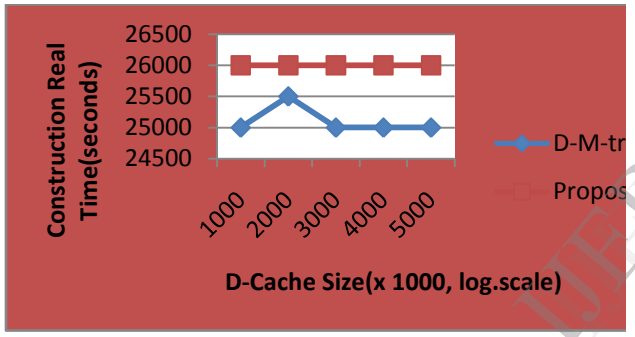


Fig. 6 - Construction of D-M-tree using single-way multi-way leaf selection.

As shown in figure 3, horizontal axis represents D-Cache size while the vertical axis represents construction real time. The results reveal that the proposed shows higher performance than the existing approach.

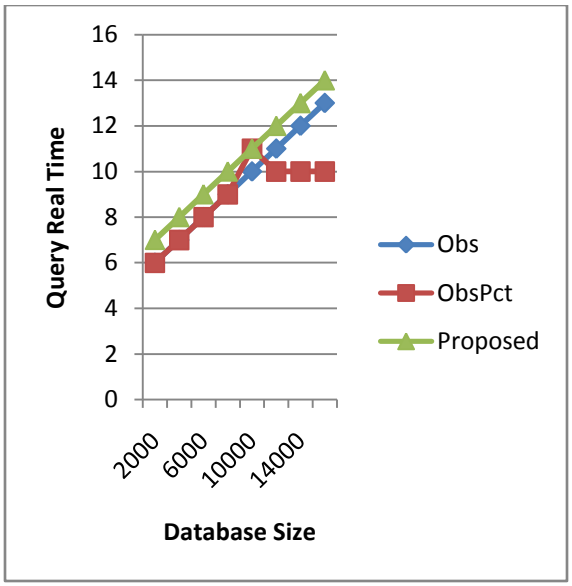


Fig. 4 - NN queries on growing database

As shown in the above figure horizontal axis represents database size while vertical axis represents query real time.With respect to NN queries against databases that grow over time, the proposedapproach outperforms existing approaches.

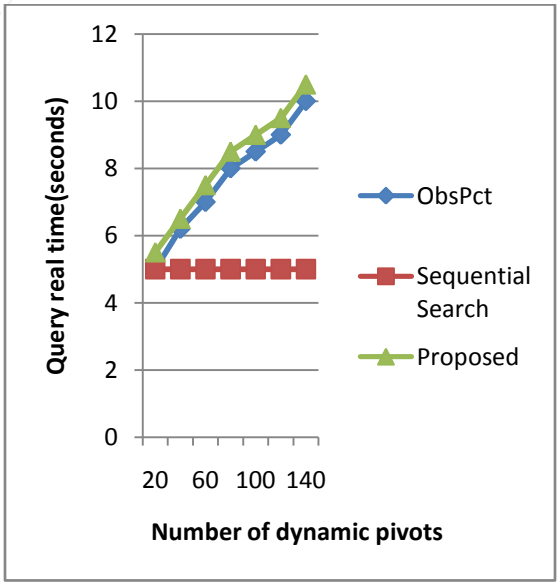


Fig. 5 - Negative results on the CoPhIR database

As shown in the above figure horizontal axis represents number of dynamic pivots while vertical axis represents query real time. The results of

proposed system are negative in this case due to the nature of database.

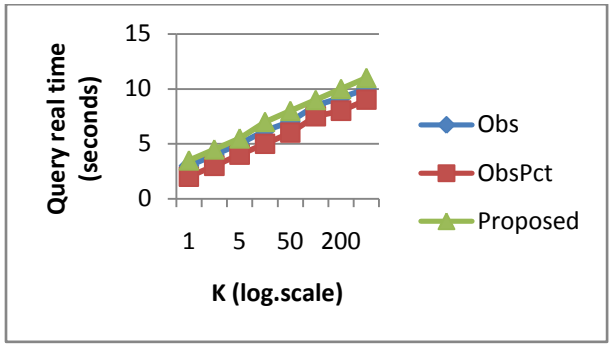


Fig. 6 - Impact of the growing number of dynamic pivots

As shown in the above figure horizontal axis represents K while vertical axis represents query real time.The results reveal that the proposed system outperforms existing approaches.

CONCLUSION AND FUTURE WORK

In this paper we studied the similarity search and its performance in the presence of D-Cache with enhanced MAMs such Pivot table, M-tree and D-file proposed by Skopal, Lokoc, and Hustos[10]. We have envisaged that the performance of similarity search can be improved further by using a metric cache in front of D-cache. By using the reusable content in both metric cache and D-cache, the proposed system is capable of improving the processing speed of similarity search. This is achieved by using metric cache that works in tandem with D-cache for higher performance. Our prototype application demonstrates the usability and performance of the hybrid approach we proposed in this paper. Empirical results are encouraging and it can be used in real world applications where similarity searches are made for metric data. One important future direction we have is to optimize the combined approach further by improving underlying MAMs. Cloud based solution is another future direction we have.

REFERENCES

[1] C. Bo¨hm, S. Berchtold, and D. Keim, "Searching in High-Dimensional Spaces—Index Structures for Improving the Performanceof Multimedia Databases," ACM Computing Surveys, vol. 33, no. 3, pp. 322-373, 2001.

[2] J.S. Vitter, "External Memory Algorithms and Data Structures:Dealing with Massive Data,"ACMComputing Surveys, vol. 33, no. 2, pp. 209-271, citeseer.ist.psu.edu/vitter01external.html, 2001.

[3] S.D. Carson, "A System for Adaptive Disk Rearrangement,"Software—Practice and Experience, vol. 20, no. 3, pp. 225-242, 1990.

[4] M. Batko, D. Novak, F. Falchi, and P. Zezula, "ScalabilityComparison of Peer-to-Peer Similarity Search Structures," FutureGeneration Computer Systems, vol. 24, no. 8, pp. 834-848, 2008.

[5] E. Cha´vez, G. Navarro, R. Baeza-Yates, and J.L. Marroquı´n,"Searching in Metric Spaces," ACM Computing Surveys, vol. 33,no. 3, pp. 273-321, 2001.

[6] P. Zezula, G. Amato, V. Dohnal, and M. Batko, Similarity Search:The Metric Space Approach (Advances in Database Systems). Springer,2005.

[7] W. Effelsberg and T. Haerder, "Principles of Database BufferManagement," ACM Trans. Database Systems, vol. 9, no. 4, pp. 560-595, 1984.

[8] H. Samet, Foundations of Multidimensional and Metric DataStructures. Morgan Kaufmann, 2006.

[9] G.R. Hjaltason and H. Samet, "Index-Driven Similarity Search inMetric Spaces," ACM Trans. Database Systems, vol. 28, no. 4,pp. 517-580, 2003.

[10] TomasSkopal, JakubLokoc and Benjamin Bustos, "D-Cache: Universal Distance Cache for Metric Access Methods". IEEE Transactions On Knowledge And Data Engineering, Vol. 24, NO. 5, MAY 2012.

[11] B. Nam, H. Andrade, and A. Sussman, "Multiple Range QueryOptimization with Distributed Cache Indexing," Proc. ACM/IEEEConf. High Performance Networking and Computing (SC '06), p. 100,2006.

[12] J.M. Shim, S.I. Song, Y.S. Min, and J.S. Yoo, "An Efficient CacheConscious Multi-Dimensional Index Structure," Proc. Int'l Conf.Computational Science and Its Applications (ICCSA '04), vol. 4,pp. 869-876, 2004.

[13] K. Kailing, H.-P.Kriegel, and M. Pfeifle, and S. Schnauer,"Extending Metric Index Structures for Efficient Range QueryProcessing," Knowledge and Information Systems, vol. 10, no. 2,pp. 211-227, 2006.

[14] J.V. den Bercken and B. Seeger, “An Evaluation of Generic BulkLoading Techniques,” Proc. 27th Int’l Conf. Very Large Data Bases(VLDB ’01), pp. 461-470, 2001.
[15] B. Braunmu¨ ller, M. Ester, H.-P.Kriegel, and J. Sander, “MultipleSimilarity Queries: A Basic DBMS Operation for Mining in MetricDatabases,” IEEE Trans. Knowledge and Data Eng., vol. 13, no. 1,pp. 79-95, Jan./Feb. 2001.
[16] R. Paredes, E. Cha´vez, K. Figueroa, and G. Navarro, “PracticalConstruction of k-Nearest Neighbor Graphs in Metric Spaces,”Proc. Fifth Int’l Workshop Experimental Algorithms (WEA ’06),pp. 85-97, 2006.
[17] F. Falchi, C. Lucchese, S. Orlando, R. Perego, and F. Rabitti,“Caching Content-Based Queries for Robust and Efficient ImageRetrieval,” Proc. 12th Int’l Conf. Extending Database Technology(EDBT ’09), pp. 780-790, 2009.
[18] F. Falchi, C. Lucchese, S. Orlando, R. Perego, and F. Rabitti, “AMetric Cache for Similarity Search,” Proc. ACM Workshop Large-Scale Distributed Systems for Information Retrieval (LSDS-IR ’08),pp. 43-50, 2008.