Available online at www.sciencedirect.com

ScienceDirect

ELSEVIER

Information Systems 32 (2007) 83–103

www.elsevier.com/locate/infosys

Data space mapping for efficient I/O in large multi-dimensional databases[☆], [☆☆]

Hakan Ferhatosmanoglu[a,*], Aravind Ramachandran[b,1], Divyakant Agrawal[c], Amr El Abbadi[c]

[a]*Computer Science and Engineering, Ohio State University, USA*
[b]*Microsoft*
[c]*Computer Science, University of California, Santa Barbara, USA*

Received 10 March 2004; received in revised form 17 May 2005; accepted 7 June 2005

Abstract

In this paper, we propose data space mapping techniques for storage and retrieval in multi-dimensional databases on multi-disk architectures. We identify the important factors for an efficient multi-disk searching of multi-dimensional data and develop secondary storage organization and retrieval techniques that directly address these factors. We especially focus on high dimensional data, where none of the current approaches are effective. In contrast to the current declustering techniques, storage techniques in this paper consider both inter- and intra-disk organization of the data. The data space is first partitioned into buckets, then the buckets are declustered to multiple disks while they are clustered in each disk. The queries are executed through bucket identification techniques that locate the pages. One of the partitioning techniques we discuss is especially practical for high dimensional data, and our disk and page allocation techniques are optimal with respect to number of I/O accesses and seek times. We provide experimental results that support our claims on two real high dimensional datasets.
© 2005 Published by Elsevier B.V.

Keywords: Data space mapping; Space partitioning; Parallel I/O; Disk and page allocation; High dimensional data; Storage; Multi-disk architectures; Performance

[☆]Recommended by F. Carino Jr.

[☆☆]This material was prepared with the support of the U.S. Department of Energy (DOE) Award No. DE-FG02-03ER25573 and National Science Foundation Award No. CNS-0403342. However, any opinions, findings, conclusions or recommendations expressed herein are those of the authors, and do not necessarily reflect the views of sponsors.

[*]Corresponding author. Tel.: +1 614 2926377.

E-mail addresses: hakan@cse.ohio-state.edu (H. Ferhatosmanoglu), araram@microsoft.com (A. Ramachandran), agrawal@cs.ucsb.edu (D. Agrawal), amr@cs.ucsb.edu (A. El Abbadi).

[1]The work was completed when the author was with the Database Lab at the Department of Computer Science and Engineering, The Ohio State University.

0306-4379/$ - see front matter © 2005 Published by Elsevier B.V.
doi:10.1016/j.is.2005.06.001

1. Introduction

The advance of technology has seen increase in applications that integrate new kinds of information, such as multimedia and scientific data. The data in these systems are usually represented by multi-dimensional summarizations of the original data. Both the dimensionality and the amount of data that need to be processed have been increasing rapidly and it becomes necessary to support the efficient retrieval of large amounts of high dimensional data. *Multimedia databases* are a typical example of such applications. Multimedia data is inherently large and is usually represented by a feature vector which describes the original data usually with a high number of dimensions. Another example of large databases of high dimensionality is *scientific databases*. Both the dimensionality and the amount of data collected and generated by scientific and engineering simulations are increasing rapidly. Some examples include climate simulation and model development, computational biology, astrophysics, high energy and nuclear physics. Many large data sets in these domains consist of a large number of attributes that may be queried and analyzed, and therefore considered as high dimensional data [1]. Satellite data repositories will soon add one to two terabytes of data in a day [2]. If the current trends continue, large organizations will have petabytes of storage managed by thousand of processors [3]. Traditional retrieval methods based on index structures developed for single disk and single processor environments [4–9] are becoming ineffective for the storage and retrieval of high dimensional data in multiprocessor and multiple disk environments. A popular approach is to develop storage and retrieval techniques that exploit I/O device and processor parallelism.

1.1. Related work

In the past, storage redundancy has been successfully exploited in the context of data declustering on multiple disks. The basic idea of current declustering approach in database management systems can be summarized as follows. First, a data space is partitioned based on a given criterion. Then the data partitions or buckets are allocated to multiple I/O devices such that neighboring partitions are allocated to different disks. Performance improvements for queries occur when the buckets involved in query processing are stored on different disks, and hence can be retrieved in parallel. Numerous methods have been proposed: disk modulo (DM) [10], fieldwise exclusive OR (FX) [11], Hilbert (HCAM) [12], near optimal declustering (NoD) [13], general multidimensional data allocation (GMDA) [14], cyclic allocation schemes [15,16], golden ratio sequences [17], Hierarchical [18], and discrepancy declustering [19] are some of the well-known disk allocation techniques. In [20–22], declustering techniques for multi-attribute databases are proposed for situations where there is some information about the query distribution. Latin squares [23] and Latin Cubes [24] have been discussed in detail for parallel access of arrays. Recently, declustering techniques have been proposed in [25] which are near-optimal for restricted cases. All of these techniques have been proposed for regular grid partitioning, where the data space is split into equi-sized partitions along each dimension. And most of them are originally proposed for two-dimensional data [26,27,15]. We have also proposed a technique for optimal declustering for two-dimensional data with a limited amount of replication [28]. There are several additional restrictions on each of them. For example, the technique in [25] requires that the number of disks is a power of a prime. FX requires that the number of disks is a power of 2. NoD was proposed only for similarity queries and requires binary partitioning in each dimension. A performance evaluation of standard declustering schemes [20,29,30] and some theoretical bounds on the cost achieved by declustering schemes [31–33] have been discussed in the literature.

For non-uniform data, the algorithms proposed for regular grid partitioning can be extended using various greedy algorithms [34,35]. Parallel R-trees [36] have been proposed as technique for parallel processing of queries. X-Trees [37] have also been proposed for indexing high dimensional data. Graph partitioning based approaches [38–40] can also be used for non-uniform data declustering. In