IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA **ITJ2574**
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

An Efficient Concurrency Control Algorithm for High-Dimensional Index Structures

Seok Il Song, Chungju National University, Korea
Jae Soo Yoo, Chungbuk National University, Korea

ABSTRACT

In this paper, we propose a concurrency control algorithm based on link-technique for high-dimensional index structures. In high dimensional index structures, search operations are generally more frequent than insert or delete operations and need to access more nodes than those in other index structures, such as B$^+$-tree, B-tree, hashing techniques, and so on, due to the properties of queries. In the proposed algorithm, we focus on minimizing the delay of search operations in all cases. It also supports concurrency control on reinsert operations for the high-dimensional index structures employing reinsert operations to improve their performance. We apply the algorithm to one of the exiting multi-dimensional index structures and implement it on a storage system. It is shown through various experiments that the proposed algorithm is more suitable for high dimensional index structures than existing ones.

Keywords: concurrency control; high dimensional index; multimedia databases.

INTRODUCTION

In the past couple of decades, multi-dimensional feature-vector-based similarity search has emerged as one of the most important research issues in the database community. Its core part is how to find the similar feature vectors from the large amount of multi-dimensional feature vectors efficiently. In order to solve the problem, research on multi-dimensional index structures has been done actively. Per the results, various multi-dimensional index structures such as R-tree (Guttman, 1984), R*-tree (Beckmann, Kornacker, Schneider, & Seeger, 1990), TV-tree (Lin, Jagadish, & Faloutsos, 1994), X-tree (Berchtold, Keim, & Kriegel, 1996), SS-tree (White & Jain, 1996), SR-tree (Katayama & Satoh, 1997), CIR-tree (Yoo et al., 1998), and Hybrid-tree (Charkrabarti & Mehrotra, 1999a) have been proposed.

In order for index structures to be used as access methods in real applications, multi-user environments must be considered. That is, there must be proper concurrency control and recovery methods for multi-dimensional index structures. Several concurrency control and recovery methods for multi-dimensional index structures were proposed such as Chen and Huang (1997), Chakrabarti and Mehrotra (1999b), Kornacker, Mohan, and Hellerstein (1997), Kornacker and Banks (1995), and Ng and Kamada (1993). Usually, multi-dimensional index structures used for similarity search systems have the following properties. First, search operations are more frequent than insert and delete operations. Second, when processing search operations, they need to access many more nodes than other index structures, such as B$^+$-Tree, B-Tree, hashing techniques, and so on, due to the characteristics of queries (Range Search, K-NN Search). Finally, some of them employ forced reinsert operations to reorganize index structures efficiently and to gain high search performance. We need to add the above properties to the design requirements of the concurrency control algorithm of multi-dimensional index structures.

In this paper, we propose a concurrency control algorithm for multi-dimensional index structures that supports reinsert operations and minimizes the delay of search operations in all cases. Also, we apply our algorithm to the CIR-Tree and implement it on MiDAS, which is the storage system of a multimedia DBMS called BADA (Chae et al., 1995). Finally, it is shown through experiments that our proposed method outperforms the existing concurrency control algorithm for GiST (CGiST) (Kornacker et al., 1997).

This paper is organized as follows. We first describe related work and the motivations of this paper, then move on to the proposed concurrency control algorithm. Next, we evaluate the performance of the proposed method and the concurrency control algorithm of CGiST through experiments. Finally, we conclude the paper.

RELATED WORK AND MOTIVATION

Multi-dimensional index structures as mentioned are in the R-tree family. They are height-balanced trees similar to the B-tree. In those index structures, leaf nodes contain index records of the form (BR, OID) where OID uniquely determines an object in the database and BR determines a bounding (hyper) rectangle of the indexed spatial object. Non-leaf nodes contain entries of the form (MBR, child-pointer), where child-pointer refers to the address of a lower node in the R-tree and MBR is the minimum bounding rectangle that contains the MBR of all of its children nodes.

Before going further, we need to mention the concepts of lathes and locks, and their compatibility matrix to make the following explanation easy. Even though latches and locks are used to maintain consistency of index trees, they are slightly different. Both of them are used to control access to shared information. Latches are like semaphores. Generally, latches are used to guarantee physical consistency of data, while locks are used to assure the logical consistency of data. Latches are usually held for a much shorter period than locks are. Also, the deadlock detector cannot recognize latch waits, so it is impossible to detect deadlocks involving latches alone, or involving latches and locks. Two lock and latch modes such as shared mode and exclusive mode are used in existing methods and our proposed algorithm. Table 1 shows the compatible matrix of locks and latches.

Several concurrency control algorithms for multi-dimensional index structures have been proposed (Chen & Huang, 1997; Chakrabarti & Mehrotra, 1999b; Kornacker et al., 1997; Kornacker &

Related Content