Optimization Algorithms for Simultaneous Multidimensional Queries in OLAP Environments

Panos Kalnis and Dimitris Papadias

Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
`http://www.cs.ust.hk/{~kalnis, ~dimitris}`

Abstract. Multi-Dimensional Expressions (MDX) provide an interface for asking several related OLAP queries simultaneously. An interesting problem is how to optimize the execution of an MDX query, given that most data warehouses maintain a set of redundant materialized views to accelerate OLAP operations. A number of greedy and approximation algorithms have been proposed for different versions of the problem. In this paper we evaluate experimentally their performance using the APB and TPC-H benchmarks, concluding that they do not scale well for realistic workloads. Motivated by this fact, we developed two novel greedy algorithms. Our algorithms construct the execution plan in a top-down manner by identifying in each step the most beneficial view, instead of finding the most promising query. We show by extensive experimentation that our methods outperform the existing ones in most cases.

1 Introduction

Data warehouses have been successfully employed for assisting decision-making by offering a global view of the enterprise data and providing mechanisms for On-Line Analytical Processing (OLAP) [CCS93]. A common technique to accelerate OLAP operations is to store some redundant data, either statically [Gupt97, GM99, SDN98] or dynamically [KR99].

Most of OLAP literature assumes that queries are sent to the system one at a time. In multi-user environments, however, many queries can be submitted concurrently. In addition, the new API proposed by Microsoft [MS] for Multi-Dimensional Expressions (MDX), which becomes de-facto standard for many products, allows the user to formulate multiple OLAP operations in a single MDX expression. For a set of OLAP queries, an optimized execution plan can be constructed to minimize the total execution time, given a set of materialized views. This is similar to the multiple query optimization problem for general SQL queries [PS88, S88, SS94, RSSB00], but due to the restricted nature of the problem, better techniques can be developed.

Zhao et al. [ZDNS98] were the first ones to deal with the problem of multiple query optimization in OLAP environments. They designed three new join operators, namely: *Shared scan for Hash-based Star Join*, *Shared Index Join* and *Shared Scan for Hash-based and Index-based Star Join.* These operators are based on common subtask sharing among the simultaneous OLAP queries. Such subtasks include the scanning of the base tables, the creation of hash tables for hash based joins, or the filtering of the base tables in the case of index based joins. Their results indicate that there are substantial savings by using these operators in relational database systems. In the same paper they propose three greedy algorithms for creating the optimized execution plan for an MDX query, using the new join operators. Liang et. al [LOY00] also present approximation algorithms for the same problem.

In this paper we use the TPC-H [TPC] and APB [APB] benchmarks in addition to a 10-dimensional synthetic database, to test the performance of the above-mentioned algorithms under a realistic workload. Our experimental results suggest that the existing algorithms do not scale well when we relax the constraints for the view selection problem. We observed that in many cases when we allowed more space for materialized views, the execution cost of the plan derived by the optimization algorithms was higher than the case where no materialization was allowed.

Motivated by this fact, we propose a novel greedy algorithm, named *Best View First* (*BVF*) that doesn't suffer from this problem. Our algorithm follows a top-down approach by trying to identify the most beneficial view in each iteration, as opposed to finding the most promising query to add to the execution plan. Although the performance of *BVF* is very good in the general case, it deteriorates when the number of materialized views is small. To avoid this, we also propose a multilevel version of *BVF* (*MBVF*). We show by extensive experimentation that our methods outperform the existing ones in most realistic cases.

The rest of the paper is organized as follows: In section 2 we introduce some basic concepts and we review the work of [ZDNS98] and [LOY00]. In section 3 we identify the drawbacks of the current approaches and in section 4 we describe our methods. Section 5 presents our experimental results while section 6 summarizes our conclusions.

2 Background

A multidimensional expression (MDX) [MS] provides a common interface for decision support applications to communicate with OLAP servers. Here we are interested on the feature of expressing several related OLAP queries with a single MDX expression. Therefore, an MDX expression can be decomposed into a set Q of group-by SQL queries. The intuition behind optimizing an MDX expression is to construct subsets of Q that share star joins, assuming a star schema for the warehouse. Usually, when the selectivity of the queries is low, hash-based star joins [Su96] are used; otherwise, the index-based star join method [OQ97] can be applied. [ZDNS98] introduced three shared join operators to perform the star joins.

The first operator is the *shared scan for hash-based star join*. Let q_1 and q_2 be two queries which can be answered by the same materialized view v. Consequently they will share some (or all) of their dimensions. Assume that both queries are non-selective, so hash-based join is used. To answer q_1 we construct hash tables for its dimensions and we probe each tuple of v against the hash tables. Observe that for q_2 we don't need to rebuild the hash tables for the common dimensions. Furthermore, only one scanning of v is necessary. Consider now that we have a set Q of queries all of which use hash-based star join and let L be the lattice of the data-cube and MV be