View Selection in DW and OLAP: A Theoretical Review

Alfredo Cuzzocrea
University of Calabria, Italy

INTRODUCTION

Data Warehousing (DW) systems store *materialized views*, *data marts* and *data cubes*, and provide nicely data exploration and analysis interfaces via *OnLine Analytical Processing* (OLAP) (Gray et al., 1997) and *Data Mining* (DM) tools and algorithms. Also, *OnLine Analytical Mining* (OLAM) (Han, 1997) integrates the previous *knowledge discovery* methodologies and offers a meaningfully convergence between OLAP and DM, thus contributing to significantly augment the power of data exploration and analysis capabilities of knowledge workers. At the storage layer, the mentioned knowledge discovery methodologies share the problem of efficiently accessing, querying and processing multidimensional data, which in turn heavily affect the performance of knowledge discovery processes at the application layer. Due to the fact that OLAP and OLAM directly process data cubes/marts, and DM is more and more encompassing methodologies that are interested to multidimensional data, the problem of *efficiently representing data cubes by means of a meaningfully selected view set* is become of relevant interest for the Data Warehousing and OLAP research community.

This problem is directly related to the analogous problem of *efficiently computing the data cube* from a given relational data source (Harinarayan et al., 1996; Agarwal et al., 1996; Sarawagi et al., 1996; Zhao et al., 1997). Given a *relational data source* $\mathcal{R}$ and a target *data cube schema* $\mathcal{W}$, the *view selection problem* in OLAP deals with how to select and materialize views from $\mathcal{R}$ in order to compute the data cube $\mathcal{A}$ defined by the schema W by optimizing both the *query processing time*, denoted by $\mathcal{TQ}$, which models the amount of time required to answer a reference query-workload on the materialized view set, and the *view maintenance time*, denoted by $\mathcal{TM}$, which models the amount of time required to maintain the materialized view set when updates occur, under a given set of constraints $\mathcal{I}$ that, without any loss of generality, can be represented by a *space bound constraint* $\mathcal{B}$ limiting the overall occupancy of the views to be materialized (i.e., $\mathcal{I} = \langle \mathcal{B} \rangle$). It has been demonstrated (Gupta, 1997; Gupta & Mumick, 2005) that this problem is *NP-hard*, thus *heuristic schemes* are necessary. Heuristics are, in turn, implemented in the vest of *greedy algorithms* (Yang et al., 1997; Kalnis et al., 2002).

In this article, we focus the attention on state-of-the-art methods for the view selection problem in Data Warehousing and OLAP, and complete our analytical contribution with a theoretical analysis of these proposals under different selected properties that nicely model spatial and temporal complexity aspects of the investigated problem.

BACKGROUND

Before going into details, in this Section we provide the conceptual basis of our work, which is mainly related to the representation of multidimensional data cubes according to the ROLAP storage model. Let $\mathcal{R} = \langle \{R_0, R_1, \ldots, R_{M-1}\}, S \rangle$ be a relational data source, such that (*i*) R_i, with $0 \leq i \leq M-1$, is a relational table of form $R_i(A_{i,0}, A_{i,1}, \ldots, A_{i,|Ri|-1})$, such that $A_{i,j}$, with $0 \leq j \leq |R_i|-1$, is an *attribute* of R_i, and (*ii*) S it the *relational schema* that models associations among relational tables in $\{R_0, R_1, \ldots, R_{M-1}\}$. Let $\mathcal{W}$ be the goal data cube schema, which, without any loss of generality, can alternatively be modeled as a *star schema*, where a central *fact table* $\mathcal{F}$ is connected to multiple *dimensional tables* $\mathcal{T}_j$, or a *snowflake schema*, where dimensional tables are also *normalized* across multiple tables (Gray et al., 1997). At the conceptual level, the goal data cube A can also be modeled as a tuple $\mathcal{A} = \langle \mathcal{D}, \mathcal{H}, \mathcal{M} \rangle$, such that: (*i*) $\mathcal{D}$ is the set of *dimensions* of $\mathcal{A}$, (*ii*) $\mathcal{H}$ is the set of *hierarchies* associated to dimensions of $\mathcal{A}$, and (*iii*) $\mathcal{M}$ is the set of *measures* of $\mathcal{A}$. Dimensions model the *perspective of analysis* of the actual OLAP model. Hierarchies are hierarchical structures (e.g., *trees*) that capture hierarchical relationships among attributes of

dimensional tables. Measures model the *analysis goals* of the actual OLAP model.

Given an N-dimensional data cube $\mathcal{A}$, and the set of dimensions $\mathcal{D} = \{d_0, d_1, \ldots, d_{N-1}\}$ of $\mathcal{A}$, all the possible (simultaneous) combinations of sub-sets of $\mathcal{D}$ define a *lattice of cuboids* (Harinarayan et al., 1996), i.e. a set of hierarchically-organized multidimensional partitions of $\mathcal{A}$. Since real-life data cube have a large number of dimensions, the resulting number of cuboids N_C is large as well (Harinarayan et al., 1996). More precisely, this number is given by the following formula: $N_C = \prod_{k=0}^{N-1}\left(2^{L_k}+1\right)$, such that L_k denotes the depth of the hierarchy H_k associated to the dimension d_k, and the unary contribution is due to the aggregation ALL. Although N_C can became prohibitively large, the cuboid lattice offers several optimization opportunities for both the two complementary problems of computing the data cube (Harinarayan et al., 1996; Agarwal et al., 1996; Sarawagi et al., 1996; Zhao et al., 1997) and selecting the views to be materialized in order to efficiently representing the data cube.

VIEW SELECTION TECHNIQUES FOR DATA WAREHOUSING AND OLAP: THE STATE-OF-THE-ART

View selection is very related to the problem of computing the data cube (Harinarayan et al., 1996; Agarwal et al., 1996; Sarawagi et al., 1996; Zhao et al., 1997), as before to materialize data cube cells, views must be selected depending on spatial and (query) time constraints. Harynarayan *et al.* (1996) first consider the problem of efficiently computing a data cube starting from the relational source. The goal is that of optimizing the query processing time, under a given space bound. To this end, Harynarayan *et al.* (1996) develop a greedy algorithm working on the cuboid lattice that tries to optimize the so-called BPUS (*Benefit Per Unit Space*) that fine-grainy models the spatial cost needed to represent materialized views. Under an optimization-oriented view of the problem, this algorithm traverses the cuboid lattice and, at each step, materializes those cuboids that, overall, give the *greatest benefit* towards improving the query processing time while lowering the BPUS. Gupta (1997) first improves this methodology with the aim of specializing it towards the proper data cube model. The result consists in introducing an elegant graph-based notion to reason on views and their underlying base relations, called AND-/OR-DAG (*Direct Acyclic Graph*). An AND-/OR-DAG is a graph such that (*i*) leafs nodes represent the base relations stored in the relational data source, (*ii*) the root represents the view to be materialized, and (*iii*) internal nodes are classified in two classes, namely AND nodes and OR nodes. AND nodes model relational algebra operations like join, projection and selection, which apply on base relations or their combinations that, in turn, model operators. OR nodes model a set of *equivalence expressions* (in terms of SQL statements generating equivalent views and intermediate views) that can be alternatively used when a choice must be done. Based on this nice theoretical model, Gupta (1997) proposes a greedy algorithm that inspects the AND-/OR-DAG in order to determine the final view set via optimizing both query processing time and view maintenance time, under a given space bound. With respect to the previous work of Harynarayan *et al.* (1996), the most significant novelty carried out by (Gupta, 1997) is represented by the amenity of considering as parameter to be optimized the view maintenance time, beyond the query processing time. Gupta and Mumick further consolidate the theory of the view selection problem with maintenance time constraint in (Gupta & Mumick, 1999), where the *maintenance-cost view-selection problem* is formalized as an extension of the baseline view selection problem. All these research results have then been synthesized in (Gupta & Mumick, 2005).

Yang *et al.* (1997) propose a set of algorithms for the view selection problem that have the particular characteristic of finding, among all the possible ones, the sub-optimal solution capable of obtaining the best *combined benefit* between two critical factors, namely the *maximization* of query performance and the *minimization* of maintenance cost. Similarly to other proposals, the main idea of this approach consists in analyzing typical queries in order to detect *common intermediate results* that can be shared among queries with the aim of reducing computational overheads, thus improving the performance.

Agrawal *et al.* (2001) demonstrate the effectiveness and the reliability of view selection tools within commercial Data Warehousing and OLAP servers in the context of the *AutoAdmin* project (Agrawal et al., 2000; Agrawal et al., 2006). In more detail, (Agrawal et al., 2000; Agrawal et al., 2006) describe algorithms (implemented within the core layer of the related tool