Materialized View Selection Using Bumble Bee Mating Optimization

Biri Arun, Jawaharlal Nehru University, New Delhi, India
T.V. Vijay Kumar, Jawaharlal Nehru University, New Delhi, India

ABSTRACT

Decision support systems (DSS) constitute one of the most crucial components of almost every corporation’s information system. Data warehouse provides the DSS with massive volumes of quality corporate data for analysis. On account of the volume of corporate data, its processing time of on-line analytical queries is huge (in hours and days). Materialized views have been used to substantially improve query performance. Nevertheless, selecting appropriate sets of materialized views is an NP-Complete problem. In this paper, a new discrete bumble bee mating inspired view selection algorithm (BBMVSA) that selects Top-K views from a multidimensional lattice has been proposed. Experimental results show that BBMVSA was able to select fairly good quality Top-K views incurring a lower TVEC. Materialization of the selected views would improve the overall data analysis of DSS and would facilitate the decision making process.

KEYWORDS

Bumble Bee Mating Optimization, Data Warehouse, Decision Support System, Materialized View Selection, Swarm Intelligence

1. INTRODUCTION

Decision making is one of the most crucial activities undertaken by individuals, business corporations, governments, etc. In order to assist managers of all levels, decision support systems (DSS) were developed, which, presently, have become one of the most essential components of almost every corporation’s information system. It is a computer system, which data managers and analysts negotiate to properly analyze corporate data and acquire the past and present business information and knowledge therein for the purpose of supporting managers in strategic decision making in real time (Sauter, 2010; Turban et al., 2005). The quality of the corporate data, which is processed by the DSS, to provide information to analysts, is of supreme importance alongside analytical models and algorithms. Poor quality of data would cause DSS to yield inaccurate, erroneous and misleading information, which when acted upon, would adversely affect corporation’s business. Corporations lose hundreds of billions of dollars annually due to poor quality of data all over the world (Turban et al., 2005).

In order to ensure high quality corporate data, data warehouse was developed. A data warehouse extracts dirty data from various transactional databases of various departments, and transforms and integrates it to produce a single uniform version of the corporate data. This is then loaded into the central database of the data warehouse (Jensen et al., 2010; Inmon, 2003). Data that enters into the data warehouse is never deleted, but is archived throughout the life of the data warehouse. Its data is never updated; instead it stores every update of the data. Unlike transactional databases, data in data

DOI: 10.4018/IJDSST.2017070101

Copyright © 2017, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
warehouse is not stored based on business processes, but is stored based on business subjects. Data warehouse is the framework that provides integrated, historical, non-volatile, subject-oriented and time-variant corporate data, which is highly reliable, to the DSS of the corporation (Jensen et al., 2010). The database of the data warehouse is designed for fast retrieval of huge volumes of data as required during data analysis. Unlike transactional databases, it stores data using the multidimensional data model in the form of de-normalized structures. The volume of its database grows with time and its tables become massive in size. On-line analytical queries posed against such massive base tables of the data warehouse require enormous amount of processing time, in hours and days, to process and extract the desired information; but its required response time is of few seconds (Gupta et al., 1997).

Increased response times, degrade on-line data analysis, due to which the extracted information may become unsuitable for making any competitive and strategic business decision. The performance of on-line analytical queries can be improved using materialized views, indexing, join indices etc. (Jensen et al., 2010). This paper focuses on the use of materialized views to improve the performance of on-line analytical queries, which in turn would facilitate better corporate decision making.

1.1. View Selection

There are two types of views, namely virtual views and materialized views. Virtual views are sub-portions of one or more base tables and are computed using costly join and aggregate operations; they are computed during runtime and are recomputed whenever their defining queries are invoked by users. With regards to data warehouse, most of the virtual views are aggregate views and hence the performance of on-line analytical queries using virtual views is very slow. Its performance can be substantially improved by using materialized views (Chirkova and Yang, 2011; Chan et al., 1999; Jensen et al., 2010). Materialized views are views that are pre-computed and stored in the database of a data warehouse. They improve query performance by causing the view defining queries to bypass its computation to directly access its pre-computed result stored in data warehouse. One time pre-computation of a materialized view saves its re-computation cost whenever its defining query is accessed. Therefore, substantial computational gain can be achieved by materializing views that are constructed using frequently posed queries (Vijay Kumar & Devi, 2012, 2013; Vijay Kumar et al., 2010a). Materialized views with weekly aggregate sales can be used to answer queries seeking weekly aggregate sales as well as monthly aggregate sales. It can also be used to answer queries that seek yearly aggregate sales. Such direct and indirect dependencies among materialized views also contribute to improve the query performance (Jensen et al., 2010). Data warehouses that use relational databases, store data using star schema; it consists of a fact table with many dimension tables. With increase in the number of dimensions of a fact table, the number of possible views increases exponentially.

Computationally, it is a challenging task to select a certain number of appropriate views from such a large number of views that would substantially improve the on-line analytical query performance. The view selection problem is shown to be an NP-Complete problem (Harinarayan et al., 1996). Selecting all views for materialization is not practical due to storage space limitations; also, a huge update cost is required to maintain all the materialized views. As discussed earlier, maintaining all views as virtual views degrades query performance; the most popular strategy to address the view selection problem is to select certain smaller sets of views which can be stored within the provided storage space and whose update costs are reasonably affordable. Harinarayan et al. (1996) studied the view selection problem and represented the problem elegantly and completely using a lattice framework. The cost of processing a view was assumed to be the number of rows in the view. Using this linear cost model, benefit measures of views were computed, based on which views were selected using a greedy algorithm. This greedy view selection algorithm of (Harinarayan et al., 1996) is hereafter referred to, in this paper, as HRUA. Gupta (1997) proposed an AND view graph to represent queries which have unique evaluation plans; a greedy algorithm was employed to select views from the AND view graph by computing the benefit of each view. An OR view graph was also proposed by (Gupta, 1997) to represent those queries that can be computed through multiple paths. Finally, an
Performance Measurement: From DEA to MOLP
www.igi-global.com/chapter/performance-measurement-dea-molp/11312?camid=4v1a

www.igi-global.com/chapter/using-systems-engineering-development-decision/75686?camid=4v1a

Improvement of the Consistent Fuzzy Preference Relation Method and Comparison with the AHP Method