Materialized View Selection using Artificial Bee Colony Optimization

Biri Arun, Jawaharlal Nehru University, School of Computer and Systems Sciences, New Delhi, India
T.V. Vijay Kumar, Jawaharlal Nehru University, School of Computer and Systems Sciences, New Delhi, India

ABSTRACT
Data warehouse is an essential component of almost every modern enterprise information system. It stores huge amount of subject-oriented, time-stamped, non-volatile and integrated data. It is highly required of the system to respond to complex online analytical queries posed against its data warehouse in seconds for efficient decision making. Optimization of online analytical query processing (OLAP) could substantially minimize delays in query response time. Materialized view is an efficient and effective OLAP query optimization technique to minimize query response time. Selecting a set of such appropriate views for materialization is referred to as view selection, which is a nontrivial task. In this regard, an Artificial Bee Colony (ABC) based view selection algorithm (ABCVSA), which has been adapted by incorporating N-point and GBFS based N-point random insertion operations, to select Top-K views from a multidimensional lattice is proposed. Experimental results show that ABCVSA performs better than the most fundamental view selection algorithm HRUA. Thus, the views selected using ABCVSA on materialization would reduce the query response time of OLAP queries and thereby aid analysts in arriving at strategic business decisions in an effective manner.

KEYWORDS
Artificial Bee Colony Optimization, Data Warehouse, Decision Making, Materialized View Selection, Swarm Intelligence

1. INTRODUCTION
Data warehouse has become an essential component of almost every information system of any enterprise. It stores subject-oriented, time-stamped, non-volatile and integrated data in the form of multidimensional data cubes to facilitate complex and fast data analysis to support decision making (Inmon, 2005; Rainardi, 2008; Choong, et al., 2007; Yu et al., 2004; Shukla et al., 1998). The size of a data warehouse and the complexity of analytical queries can significantly delay query response time. It requires days to answer complex analytical queries posed directly against the base dimension tables of a data cube implemented using star schema in a data warehouse; but the general requirement for Online Analytical Processing (OLAP) query execution time is of few seconds or minutes (Chirkova et al., 2001; Chaudhuri and Dayal, 1997; Gupta, 1997; Agarwal et al., 1996; Kumar et al., 2006). With increased query execution time, the availability of information, for strategic and tactical business decision making in real time, is greatly reduced. OLAP query execution time can be greatly reduced by query optimizers, query evaluation techniques, indexing strategies, materialized views etc. to leverage decision making (Harinarayan et al., 1996; Gyssens and Lakshmanan, 1997; Niemi...
et al., 2001; Gray et al., 1996; Golfarelli et al., 2004). This paper focuses on the use of materialized views to improve OLAP query execution time in a data warehouse environment. An analytical query defines a view, in whose context it analyzes a fact for arriving at business decisions (Shukla et al., 1998; Shukla et al., 2000). A view whose fact (aggregate) has been pre-computed and substantiated is a materialized view (Kotidis, 2002). OLAP queries involving computationally expensive joins and aggregations are generally materialized to improve query performance (Chaudhuri and Dayal, 1997; Shukla et al., 1998; Yang et al., 1997). The challenges associated with materialized views are: identification of views for materialization, efficient computation of selected views from base tables, using materialized views to answer queries and efficient updating of materialized views on arrival of new data into data warehouse (Golfarelli et al., 2004; Gupta et al., 1997; Gupta and Mumick, 2005).

Some computationally expensive queries are posed more frequently than others; pre-computing and materializing such queries would enable the system to respond quickly. Furthermore, some queries may help in answering many other queries and may improve the system’s query performance. Identifying a set of such queries over a database schema under a workload of queries and storage space constraint is referred to as view selection (Chaudhuri and Dayal, 1997; Shukla et al., 1998).

Selecting the entire set of all possible views for materialization would be one way of view selection, but this wouldn’t be feasible for higher dimensional data sets. The set of all possible views grows exponentially with the increase in the number of dimensions and hence, it would be impossible to store them with limited disk space. The other extreme would be to select none of the views, but to compute them in real time; this is also infeasible due to huge delay in query response time on account of computing expensive join and aggregation operations. A feasible approach would be to select and materialize a set of beneficial views that satisfies disk space, maintenance cost and query response time constraints (Chan et al., 1999). The problem of selecting such an optimal subset of views is shown to be an NP-complete problem (Harinarayan et al., 1996). Alternatively, these views can be selected empirically (Aouiche et al., 2006; Vijay Kumar & Devi, 2012, 2013; Vijay Kumar et al., 2010a) or based on heuristics like greedy (Harinarayan et al., 1996; Nadeau & Teorey, 2002; Haider & Vijay Kumar, 2011, 2017; Vijay Kumar, 2013; Vijay Kumar & Ghoshal, 2009; Vijay Kumar & Haider, 2010, 2011a, 2011b, 2012, 2015; Vijay Kumar et al., 2010b, 2011), randomized (Derakhshan et al., 2006; Derakhshan et al., 2008; Vijay Kumar & Kumar, 2012a, 2012c, 2015), evolutionary (Horng et al., 1999; Zhang et al., 1999, 2001; Lee & Hammer, 2001; Yu et al., 2003; Lawrence, 2006; Vijay Kumar & Kumar, 2012b, 2013, 2014), swarm based (Sun & Wang, 2009; Arun & Vijay Kumar, 2015a, 2015b, 2017; Vijay Kumar & Arun, 2016, 2017). Among these, the greedy view selection algorithm given in (Harinarayan et al., 1996) is considered the most fundamental view selection algorithm, which hereafter in this paper would be referred to as HRUA. HRUA greedily selects Top-K views from a multidimensional lattice. HRUA’s execution time is exponential with respect to the number of dimensions of fact table and therefore, it fails to compute near optimal sets of views for fact tables with higher dimensions. Stochastic search optimization techniques have been shown to be more flexible, robust, efficient and scalable and one such class of techniques, i.e. swarm intelligence, can be used to address the view selection problem. In this paper, an attempt has been made to address the view selection problem using Artificial Bee Colony (ABC) algorithm. ABC is a biologically inspired swarm intelligence technique based on the collective and cooperative foraging behavior of bee swarms (Karaboga, 2005: Bolaji et al., 2013). The bees of a bee swarm are divided into queen bee, drones and workers. Only the worker bees are engaged in foraging. The worker bees are further divided into employed bee, scout bee and onlooker bee. The worker bees communicate and cooperate with each other through dance languages to maximize foraging while minimizing their energy expenditure (Karaboga, 2005). The ABC algorithm mimics the communication and cooperative behaviors of these bees during foraging to solve complex optimization problems. The ABC algorithm
22 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product's webpage:

www.igi-global.com/article/materialized-view-selection-using-artificial-bee-colony-optimization/175327?camid=4v1


www.igi-global.com/e-resources/library-recommendation/?id=2

Related Content

Investigation into Factors that Influence the Use of the Web in Knowledge-Intensive Environments
Yong Jin Kim, H. Raghav Rao and Abhijit Chaudhury (2002). Intelligent Support Systems: Knowledge Management (pp. 135-144).
www.igi-global.com/chapter/investigation-into-factors-influence-use/24449?camid=4v1a

Using Ambient Social Reminders to Stay in Touch with Friends
www.igi-global.com/chapter/using-ambient-social-reminders-stay/53336?camid=4v1a
KStore: A Dynamic Meta-Knowledge Repository for Intelligent BI
[www.igi-global.com/article/kstore-dynamic-meta-knowledge-repository/2452?camid=4v1a](www.igi-global.com/article/kstore-dynamic-meta-knowledge-repository/2452?camid=4v1a)

Building Textual OLAP Cubes Using Real-Time Intelligent Heterogeneous Approach
[www.igi-global.com/article/building-textual-olap-cubes-using-real-time-intelligent-heterogeneous-approach/204954?camid=4v1a](www.igi-global.com/article/building-textual-olap-cubes-using-real-time-intelligent-heterogeneous-approach/204954?camid=4v1a)