Exploiting DBMS Idleness Periods for Tuning its Performance

AbdulMateen, Basit Raza[1], Muhammad Sher[2], Mian Muhammad Awais[3]
[1]Computer Science Department, Federal Urdu University of Arts, Science & Technology,
Islamabad, Pakistan
[2]Computer Science Department, International Islamic University,
Islamabad, Pakistan
[3]Computer Science Department, Lahore University of Management Sciences,
Lahore, Pakistan
abdulmateen@fuuastisb.edu.pk, basitrazasaeed@gmail.com, m.sher@iiu.edu.pk, awais@lums.edu.pk

ABSTRACT: *Rapid growth in data, maximum functionality and changing behavior tends the database workload to be more complex and tricky. Every organization has complex types of database workloads that are difficult to manage by the humans; human experts take much time to get sufficient experience to manage workload efficiently; even in some cases this task becomes impossible and leads towards malnourishment. The versatility in database workload leads practitioners and researchers towards new challenges. The paper presents a novel approach for the workload management through Case Base Reasoning characterization approach, impact based scheduling and DBMS idleness detection. The approach will be validated through experiments which will be performed over the real time workload to reveal the efficiency and accuracy. The ultimate goal of the research is to provide an Autonomous Workload Management framework with characterization, scheduler and idleness detection modules.*

Keywords: Workload, Autonomous, Characterization, Scheduling, Idleness Detection

Received: 11 November 2013, Revised 8 December 2013, Accepted 13 December 2013

1. Introduction

The complexity in Database Management System (DBMS) increases due to the various functionality demands from the users, complex data types, diverse workload and data volume that is increasing with the passage of time. These factors cause brittleness and unmanageability in DBMSs. To handle these problems organizations hire number of expert Database Administrators (DBA) and spending lot of money to get expected improvement, throughput and response. Usually DBA have to take care of all the database tasks such as making policy for workload priorities, memory and storage configuration and such other tasks. The cost of hardware is decreasing but the cost of management is increasing. Performing workload management activities manually, by hiring experts causes increase in Total Cost of Ownership (TCO) [1-3]. Moreover with the advent of distributed systems and Data Warehouse (DW), it became difficult and even some cases impossible for DBA to manually organize, optimize and configure day to day database tasks. For efficient workload management, queries may be stopped for a while and later on restarted. However, when queries will be stopped during their execution then executed part will be either lost or have to be saved in memory for further use.

The style of the database workload has been changed from the offline analysis to online adaptation. In workload management there are three units which are workload, resources and objectives. All three workload units are co-related with each other. The workload uses some resources to meet the objectives of an organization or we can say resources are allocated through different approaches to workload which has some management objectives. Workload has evolved through three phases which are capacity planning, resource sharing and performance oriented workload [4]. Capacity planning workload management is based on cost sharing; the idea behind the resource oriented approach is maximum resource utilization while in case of performance oriented approach the focus is on the business goals and objectives. In workload management, the main functions are workload frequency patterns, composition, intensity and required resources.

The proposed approach will be used to identify the workload classes by partitioning the global workload into smaller sets composed of transactions or requests. The proposed technique will be validated through various experiments over the real-time industrial and benchmark workloads. These experiments will be performed on MySQL database. According to our literature survey, previous workload management approaches take extra time for characterizing the workload while our proposed approach does not take any time for the characterization of database workload. The proposed research will be a milestone towards the Autonomous Workload Management (AWM) for database management systems and data warehouses.

Rest of the paper is organized as: Section 2 presents the state of the art that is related with database workload management techniques. Section 3 elaborates our proposed workload management approach. Section 4 is dedicated for the experimental setup and results validation. Section 5 concludes the research with some key future directions.

2. Related Work

The research on workload characterization and management in DBMSs and DWs has been carried out from the last many years. We surveyed a large number of white papers, research papers and case studies that are related with the DBMS and DW characterization and management. In this section, we are presenting some of the previous research on workload characterization and management techniques.

Zewdu et. al. [5] proposed a model to identify and characterize the workload; and identified the variables that affect the workload. Workload classification into Online Transaction Processing (OLTP) and Decision Support System (DSS) is performed through two algorithms, i.e. Hierarchical clustering and Classification & Regression Tree (CRT). Degree of correlation is determined through Hierarchical clustering. They identified status variables that affect the workload and use them to measure the workload cost. The proposed approach is validated by analyzing and experimented over TPC workload benchmark queries and transactions in MySQL 5.1 database.

After performing number of experiments they concluded that four variables (Com_ratio, nnodb_log_writes, Qcache_hits and Questions) are more useful for the workload classification in MySQL.

Elnaffar et. al. [6] proposed a workload characterization model that automatically classifies the workload into OLTP and DSS. The model works on the basis of workload characteristics and uses the browsing and ordering profiles of TPC-W benchmark. The proposed classification model is validated through number of experiments that are performed by taking the benchmark workloads.

Menasce et. al. [7] introduces a characterization technique that generates workload models for E-commerce. A Customer Behavior Model Graph (CBMG) or state transition graph is introduced that represents similar navigational pattern for group of customers who perform same activities. The workload model with related parameters is identified and workload characterization is performed through clustering algorithm. The technique is evaluated by executing number of experiments. The proposed technique has some problems, for example there will be maximum session drops during the huge sessions or maximum load and there is no mechanism defined in the proposed research to manage or recover these session drops.

REDWAR (RElational Database Workload AnalyzeR) [8] is an offline tool for DB2 environment, which is used for the characterization of SQL trace and provides the structural information with statistics of the query under execution. REDWAR analyzes and classifies the data. During analysis, it uses statistical summaries (correlation, distribution, variation) and runtime behavior of the workload. The report generated by the REDWAR can be used to plan the physical design and build benchmark workload. It increases efficiency of the database system by identifying the criteria for a query.

Wasserman et. al. [9] presented an analysis of the characterization technique for Business Intelligence (BI) workload. The

proposed approach is based on some resource related attributes such as CPU consumption, sequential and random I/O rate and joins degree. The proposed characterization approach is performed on the TPC-H benchmark and mostly parameter values are based on the assumptions. The proposed characterization technique is limited as it considers only one parameter that is user resource demand.

We have also reviewed some other workload management techniques, models and frameworks such as Surajit et al. [10, 11] proposed a framework to stop and restart the queries during their execution to manage the workload efficiently. The restart approach re-executes the stopped query from the position where it was stopped. This technique does not save all the executed information but save only the selected information from the past execution to reduce memory overhead. This method also reduces the running time of re-executed queries. The proposed technique is validated by making experiment over real and benchmark data. However, the proposed approach is limited to handle single query execution plan (QEP) and does not consider parallel QEPs. There is no dynamic way in proposed technique to maintain the restart plan during the modification of source records (past executed records).

Mumtaz et. al. [12-14] discussed the impact of query interaction over the workload by using planning experiment and statistical modeling approach. This approach is portable as it works without previous assumption about the internal knowledge of database and the cause of query interaction. A framework Query Shuffler (QShuffler) – Interaction aware query scheduler for workload management in Business Intelligence is introduced. QShuffler considers the impact of queries over each other as it has vital role in performance. Different requests are given by the users in the form of queries; QShuffler classifies these queries according to their type and arranges them in a way to minimize the total execution time of the workload. The QShuffler is implemented in IBM DB2 and evaluated with TPC-H workload. It gives four times performance over the FCFS scheduler of database systems. However in QShuffler, the large jobs have to wait for a long time even these are of higher priority as it uses SJF algorithm for scheduling. The average execution time is larger as it uses non-preemptive SJF approach.

Krompass et al. [15-17] introduced an adaptive Quality of Service (QoS) management technique, in which economic model is used to handle individual request of BI and OLTP workload proactively. They provide a systematic way to arrange different requests by dividing these into different classes based on cost and time limit. The framework is evaluated to observe its effectiveness by performing experiments on different workloads. The framework uses economic model with two economic cost functions (Opportunity Cost, Marginal gains) where penalty information is added with the queries. The scheduling policy used for OLTP workload in framework is enhanced by considering the combine effect of priorities and SLOs rather than considering merely priority.

A technique for query suspension and resumption [18, 19] with minimum overhead is discussed where the author proposed induction of asynchronous checkpoints for each cardinality in a query. Mehta et. al. [20] define the design criteria that make a Mixed Workload Scheduler (MWS) and uses it to design rFEED, i.e. MWS that is Fair, Effective, Efficient, and Differentiated. BI workload manager [21, 22] for enterprise data warehouse is introduced in which queries are admitted in the form of batches. Pang et al. devices a Priority Adaptation Query Resource Scheduling (PAQRS) Algorithm [23] based on the Priority Memory Management (PMM) Algorithm and deals with multi-class query workload.

The previous characterization and management techniques do not identified the parameters such as CPU or execution time, response time, arrival time, memory, I/O cost, communication times. Most of the previous techniques are based on assumptions or taking values from the query optimizers rather than calculating the actual values. However, in reality there is a huge difference between the guessed and actual values. None of the previous characterization technique for workload management in DBMSs identified the OLTP or OLAP workload using such novel approach. The research is in progress and currently we are developing the characterization, scheduler and idleness detector module of the AWM. The work is followed by testing the proposed framework over the various real-time industrial and benchmark workload. There is a need of workload manager that can manage the DBMS or DW workload without affecting other requests, efficient resources utilization and handle all other related matters.

3. Proposed Solution

Database workload consists of SQL statements, transactions, scripts, commands, sessions and Jobs. Different researchers, practitioners and vendors have been working on workload management in DBMSs and DWs. In this regard, various techniques have been introduced to mange workload that includes Linear Programming, Mixed- Integer Programming, Queuing Theoretic Model, Largest Memory Priority (LMP), Priority Gradient Multiprogramming (PGM), and Shortest Job First (SJF) algorithm.

However, these techniques have certain limitations: the workload optimization techniques that are using non-preemptive scheduling scheme provide inefficient results due to their poor responsiveness that may lead to starvation and hanging; most of the previous techniques use approximate values rather than calculating actual values; considering all the queries at same level; dividing the workload into batches or taking multiprogramming (MPL) level on the basis of single or two parameters. By taking the approximate values, considering all queries at same level, avoiding query interaction and fixed query types lead to inefficient results. Dividing batches or MPL setting on the basis of single parameter is not a sufficient criterion to achieve maximum efficiency. Our proposed solution consists of three main modules which are the Workload Characterization, Scheduler and Idleness Detector as shown in Figure 1.

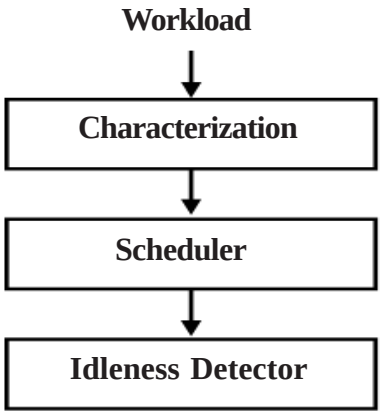

Figure 1. Modules of AWM Framework

3.1 Characterization

The responsibility of the Characterization module is to analyze the incoming workload and observe each query with respect to some key parameters like how much memory is required, what will be the cost of the query, priority and its dependency (with other queries). Steps of the proposed characterization approach are as under:

The incoming workload will be characterized into OLTP and Online Analytical Processing (OLAP) on the basis of some key attributes of the workload. In MySQL, there are 291 status variables that vary during the workload execution. From these 291 status variables, we will select those which can be used to identify the particular workload type. The steps of the proposed approach are shown in Figure 2. We will develop a Case Base Reasoning (CBR) that contains the effective workload parameters with their associative workload type (OLTP or OLAP). To develop the training data, effective parameters will be listed with their associated workload type. The technique will work by extracting the above parameters for each query of the incoming workload and values of these parameters are searched in the CBR. The query will characterize as OLTP or OLAP when exact or nearest match against the parameters is found. However in case of unsuccessful search, Bayesian classification approach will be used to decide about the query type, where historical data is used to make decision about workload type. After the identification of the workload type it will be stored in the CBR as a new case for future use.

The proposed technique for characterization of database workload will work as the user write query in the workload editor. The characterizing process will work in parallel with user input to workload editor. So the workload will be characterized and classified before the start of workload execution. In previous characterization techniques, the characterizing process starts after the submission of workload to system. In contrast to the previous approaches, our approach will characterize the workload in advance. Due to this reason, our proposed characterizing technique will save the precious execution time that ultimately manages the workload in more efficient manner.

3.2 Scheduler

The Scheduler will take the database workload and calculates their related impact. The impact of the workload will be calculated on the basis of some main parameters (CPU or execution time, response time, arrival time, memory, I/O cost, communication times) and workload type (OLTP, OLAP) as identified in the characterization step. After calculating the impact of each workload, a comparison of the workload impact will be performed. On the basis of comparison all the workloads will be arrange.

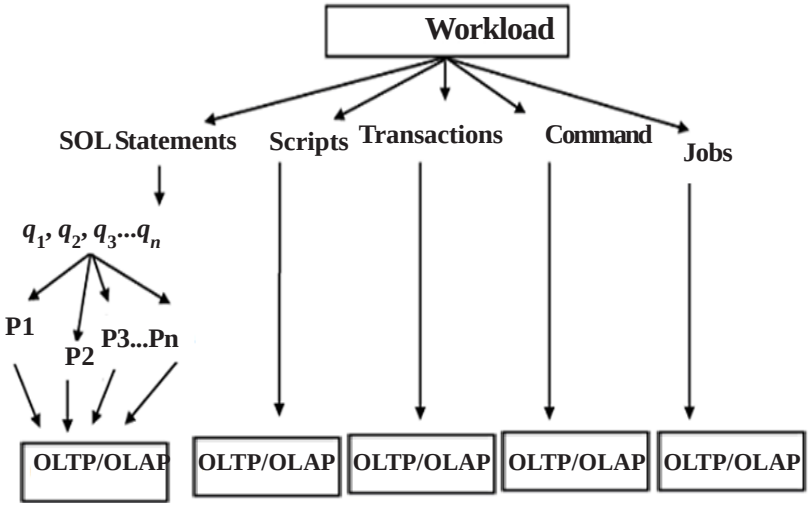

Figure 2. Steps to Characterize the DBMS/DW Workload

3.3 Idleness Detection

There will be a CBR that will have initially training data on the basis of important workload parameters. These parameters include Workload id, Number of OLTP queries, Number of OLAP queries and idleness percentage. These parameters will be listed with their associated Idleness. The idleness of the incoming workload will be identified by searching the exact or nearest (almost equal) workload in the CBR. After getting the exact or nearest match, idleness of that workload will be taken from the CBR. However if the particular workload is not found then it will be executed without considering the idleness for the first time and after its execution it will be stored as a new case in CBR with associated idleness. After the detection of idleness, this time period will be used for de-fragmenting the data, index reorganization or any other database activity. The de-fragmenting or other processes will execute up to the detected idleness. After the completion of identified idleness time, these processes either may complete their execution or suspend.

4. Experimental Setup and Results Validation

Hardware that will be used to establish the experimental setup includes Database Server, Backup Server and five client computers. All the devices are connected through a network as shown in Figure 3. We have selected the MySQL Database Management System to execute different benchmark and industrial workload to validate our workload characterization and management framework. The selection of MySQL is due to its free availability and open source feature. Each client machine will have the ability to make thousands of queries to database server at a time. These queries are generated via threads at client machines and send to database server.

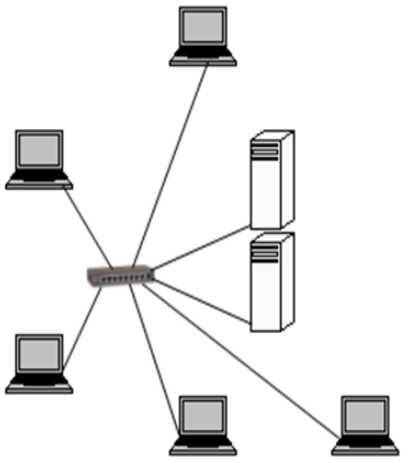

Figure 3. AWM Experimental Setup

The proposed technique will be validated by observing two parameters, i.e. Average wait time and CPU utilization. Values of these parameters will be recorded by executing the workload. Comparison reports and graphs will be generated to show the efficiency and accuracy of our proposed technique. Moreover, we will also calculate the overheads and expenses of the proposed solution.

5. Conclusion & Future Work

The research proposed a novel workload management framework that makes decision about the workload execution. The research over the Autonomous Workload Manager is in progress and after the completion initial design; we are now working on the implementation of the characterization, scheduler and idleness detector modules. The proposed research is different from the previous work in number of ways. Our work contributes by analyzing the various existing characterization and scheduling techniques that are used for workload management in DBMSs and DWs. The selection of the different workload characterization parameters will be identified and on the basis of these parameters workload is identified as OLTP and OLAP. A new scheduling technique will be introduced where workload will be arranged according to their impact and finally DBMS idleness will be detected to improve the performance.

References

[1] Huebscher, M. C., McCann, J. A. (2008). A survey of Autonomic Computing—Degrees, Models, and Applications, ACM 0360-0300/2008/08-ART7.

[2] Manish Parashar., Salim Hariri. (2005). Autonomic Computing: An Overview, Springer-Verlag Berlin Heidelberg, LNCS 3566, p. 247–259.

[3] Sam, S., Lightstone., Guy Lohman., Danny Zilio. (2002). Toward Autonomic Computing with DB2 Universal Database, SIGMOD, 31(3).

[4] Niu, B., Martin, P., Powley, W., Horman, R., Bird, P. (2006). Workload Adaptation in Autonomic DBMs, Proc. of CASCON2006, Canada, p. 161-173.

[5] Zewdu, Z., Denko, M. K., Libsie, M. (2009). Workload Characterization of Autonomic DBMSs Using Statistical and Data Mining Techniques, AINA 2009, p. 244-249.

[6] Elnaffar, S., Martin, P., Horman, R. (2002). Automatically classifying database workloads, *In*: Proceedings of the CIKM'02, Washington, DC, November 2002.

[7] Menasce, D. A., Almeida, V. A. F., Fonseca, R., Mendes, M. A. (1999). A Methodology for Workload Characterization of E-commerce Sites, Proceedings of the 1st ACM Conf. on Electronic Commerce, USA, p. 119 – 128.

[8] Yu, P., Chen, M. S., Heiss, H.-U., Lee, S. (1992). On Workload Characterization of Relational Database Environments", IEEE Transactions on Software Engineering, 18(4), p. 347–355.

[9] Wasserman, T. J., Martin, P., Skillicorn, D. B., Rizvi, H. (2004). Developing a characterization of business intelligence workloads for sizing new database systems, 7th ACM Workshop on Data Warehousing & OLAP, 2004, 7–13.

[10] Surajit, C., Raghav, K., Ravishankar, R., Abhijit, P. (2007). Stop-and-Restart Style Execution for Long Running Decision Support Queries, VLDB, p 735-745.

[11] Chaudhuri, S., Kaushik, R. and R. Ramamurthy, When Can We Trust Progress Estimators For SQL Queries?, ACM SIGMOD, 2005, pp. 575–586.

[12] Mumtaz, A., Ashraf, A., Shivnath, B. (2008). Modeling and Exploiting Query Interactions in Database Systems, CIKM'08, California, USA.

[13] Mumtaz, A., Ashraf, A., Shivnath, B., Munagala, K. (2008). QShuffler: Getting the Query Mix Right, International Conference on Data Engineering (ICDE), p. 1415-1417.

[14] Mumtaz, A., Ashraf, A., Shivnath, B. (2009). Query interactions in database workloads, DBTest.

[15] Stefan, K., Andreas, S., Martina-Cezara, A., Harumi, K., Janet, W., Umeshwar, D., Alfons, K. (2008). Quality of Service Enabled Management of Database Workload, In Service-Oriented Computing – IEEE Computer Society Technical Committee on Data Engineering.

[16] Dayal, U., Kuno, H. A., Wiener, J. L., Wilkinson, K., Ganapathi, A., Stefan, K.(2009). Managing operational business intelligence workloads. Operating Systems Review, 43 (1), p. 92-98.

[17] Krompass, S., Kuno, H. A., Wiener, J. L., Wilkinson, K., Umeshwar, D. (2009). A Testbed for Managing Dynamic Mixed Workloads, PVLDB 2(2), p. 1562-1565.

[18] Chandramouli, B., Bond, C. N., Babu, S., Yang, J. (2007). Query Suspend and Resume, *In*: Proceedings of the ACM SIGMOD International Conference on Management of Data.

[19] Chandramouli, B., Christopher Bond, N., Shivnath, B., Jun, Y. (2007). On Suspending and Resuming Dataflows, International Conference on Data Engineering (ICDE), 2007, p. 1289-1291.

[20] Abhay, M., Chetan, G ., Song, W., Umeshwar, D. (2009). rFEED: A Mixed Workload Scheduler for Enterprise Data Warehouses, International Conference on Data Engineering (ICDE), p.1455-1458.

[21] Mehta, A., Gupta, C., Dayal, U. (2008). BI Batch Manager: A system for managing batch workloads on enterprise data warehouses, EDBT 08.

[22] Mehta, A., Gupta, C., Song Wang., Dayal, U. (2008). Automated Workload Management for Enterprise Data Warehouses, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering.

[23] Pang, H. H., Carey, M. J., Livny, M. (1995). Multiclass Query Scheduling in Real-Time Database Systems, *IEEE Transactions on Knowledge and Data Engineering*, 7 (4), ISSN:1041-4347, p. 533-551.