Self-Prediction of Performance Metrics for the Database Management System Workload

Basit Raza, Abdul Mateen, Muhammad Sher, and Mian Muhammad Awais

***Abstract*—Workload in Database Management System (DBMS) consists of huge amount of data and number of concurrent users who are executing different requests that require some resources. To manage these types of activities, organizations hire different database experts. There is versatility in workload due to the huge data size and different types of requests (workload). These factors contribute to some new challenges in the workload management. These challenges are identification of the workload and decision about the problem queries, identification of resource oriented and contention queries, accurate workload classification, optimal plan selection, prediction and adoption. In DBMS, where workload management and tuning is performed through if-then approach, unforeseen behavior of the workload cannot be handled and sometime leads to unpredictable state. In this research a prediction framework has been proposed called as workload queries performance Predictor. The predictor will predict the performance metrics (workload size, elapsed time, record accessed, record used, disk I/Os, memory required, message count and bytes) for queries in a given workload. We are improving efficiency and reducing search time when projection of query feature vector is performed over performance feature vector. The predictor will take help from the optimizer and store the information in database which saves the information as history for the future.**

***Index Terms*—Workload management, prediction, performance metrics**

I. Introduction

The complexity and heterogeneity of computer systems increasing day by day, which leads towards development of autonomic computing (AC) systems. AC system has five stages or levels of autonomicity which are Basic, Managed, Predictive, Adaptive and Autonomic [1, 19]. There are number of challenges in autonomic computing that started from first level to last level. These challenges are conceptual, architecture, middleware and applications challenges [2]. There are five components of Autonomic computing systems, which are Negotiation, Execution, Observation, Deliberation and Failure Recovery [3]. The AC benefits are also important in DBMSs. The factors that motivate the incorporation of AC in DBMSs are data volume growth and increase in user's functionality requirement and shortage of skillful database administrators (DBAs). AC has different characteristics (self-*) [4-6] which are Self-Optimization, Self-Configuration, Self-Healing, Self-Protection Self-Inspection, Self-Organization, Self-Prediction and Self-Adoption. Self-Prediction is a characteristic in which a system has the ability to predict for the future according to available resources and environment. Self-Adoption is a characteristic of autonomic system to adopt the changes dynamically to improve the system performance.

DBMS is the main source of information in any organization. Accurate and timely information leads towards success of the organization. In DBMSs workload consists of batch, incremental data loads, batch reports and complex requests encountered in to the system. These requests may be related with insertion, deletion and updation of data and also demanding the resources which can be memory, Input/output devices or others etc. The type of workload is an important factor in DBMSs, which require serious attention. Workload may be of any type it may be either Online Transaction Processing (OLTP) or Decision Support System (DSS) or Online Analytical Processing (OLAP) [17 - 18]. Resources are allocated by considering the type of the workload. DBA was responsible to manage workload by manually tuning and configuring the workload to improve performance. Now due to the growth in data volume and changing behavior of the workload it is beyond the human capability to manage such workload. In normal routine DBMSs perform their function in a managed way. But sometimes DBMSs become overloaded due to less number of resources for large number of incoming requests or mismanagement of available resources etc. When numbers of requests are greater than the requests that a workload manger can handle then either the next request will be rejected or will be attended when it will become free. The workload manager that cannot deliver it services due to any reason then the DBMS becomes down and losing the interest of user, wasting the time and money.

The systems that monitor all the time and provide accurate predictions about the workload change or resource demand are called self-predicting systems. These systems are dependent on identification of workload that provides the workload information. On the basis of this information, previous history and mathematical models, self-predicting system predict for the future. The system predicts itself about tuning, acquisition planning and resource allocation by reducing the detailed workload and internal system knowledge required [20]. Self-prediction is the characteristic of DBMS to predict the future on the basis of historic and statistical data. Wealth of research has been done in the context of workload prediction.

Manuscript received February 19, 2012; revised March 30, 2012.

Basit Raza, Abdul Mateen are with the Computer Science scholar at International Islamic University, Islamabad, Pakistan (e-mail: basitrazasaeed@gmail.com, abdulmateen@fuuastisb.edu.pk).

Mian Muhammad Awais is with the Department of Computer Science, Lahore University of Management Sciences, Lahore, Pakistan (e-mail: awais@lums.edu.pk).

Muhammad Sher is with the Department of Computer Science and Software Engineering, International Islamic University, Islamabad, Pakistan (e-mail: m.sher@iiu.edu.pk).

The organization of the paper is as followed, section II provides the literature review in the area of self-prediction with limitations. Section III introduces the proposed methodology for prediction the performance metrics of the given workload. Section IV provides the conclusion of the research with future direction.

II. Literature Review

Chetan Gupta et. al. [7, 8] proposed a predicting model for the query execution time in a warehouse environment. The proposed model uses query execution plan (QEP) and the load on the system to predict the query execution time through machine learning techniques. The approach is validated and can be incorporated into commercial DBMSs. The model predicts on the basis of historical data in binary tree like structure which builds a tree so called as Predictions of Query Runtime (PQR) tree. The node of the binary tree depicts the time ranges and the ranges of child nodes do not overlap and it is subset of the time ranges of parent node. These trees are developed through machine learning approach. These trees build in two steps, which are obtaining a PQR tree and time ranges of incoming query. After constructing the tree, it is being applied and updation is made periodically. Finally, the execution time of workload is predicted. There some limitation in the model as it does not consider the different time spans of the day, i.e. at some time workload is low or high in frequency. Further it does not handle sudden changing behavior of the workload.

Dayal et. al. [9] evaluated the existing algorithms for long running queries and introduces an approach to manage workload through resource usage prediction of queries. The approach uses Kernel Canonical Correlation Analysis (KCCA) which identifies the correlation between query properties and performance attributes on the training data. On the basis of statistical relationship it predicts the performance of incoming queries. Query features are identified through machine learning algorithms and calculate similarities between the pairs of query feature vectors. The identified query features are used by the KCCA model to search its coordinates for mapping on query projection and performance projection. The approach uses K-Nearest neighbor algorithm. The required predicted metrics are achieved through mapping of these metrics with performance projection. The KCCA model adopted by the prediction framework does not predict all queries. The KCCA model does not have the ability to perform continuous retraining.

Ganapathi et. al. [10] proposed a framework and developed a system to predict the performance metrics such as elapsed time of query, records used, disk input/output and message bytes for the queries. The proposed model uses statistical machine learning approach. The prediction is performed on customer data and the data generated from some system using machine-learning techniques. The proposed technique provides accurate results and predicts these metrics simultaneously using the information that is available before the query execution. The framework is validated by performing experiments in HP Neoview database and it is found that the predicted elapsed time for most of the queries remain around 20 percent of the actual elapsed time. The framework also provides information about the short and long running queries. The framework is tested for the queries that have 10 minutes execution time for 32-node system. The prediction framework predicts 85% of the total queries accurately (20 percent margin).

Said Elnaffar and Martin [11, 21, 22] have proposed a framework Psychic-Skeptic Prediction Framework (PSP), which is used to predict workload shifts when it is due from decision support (DSS) to other type of workload i.e. OLTP.

PSP framework predicts the workload shift through offline and online strategy. The proposed framework works by classifying the workload and handle the workload without any human involvement. The PSP framework consists of three components which are Training Data Model, Psychic (offline) and Skeptic (online). Psychic uses polynomial regression technique on consolidated scenario and builds an offline prediction model. The Skeptic component uses linear model to verify trends of the shifts. The PSP is offline however when it estimates the time interval for expected shifts using historical workload models it works online. During online prediction whenever the workload shift is found, the Skeptic component performs short-term prediction through linear regression method. As compare to other online prediction techniques it has less overhead. This model recommends the best mode for certain environment by differentiating the estimates of best and worst performance. Along with automatic predictability this methodology may be helpful for many other operations such as index rebuilding, statistics updation and reorganization of data. The PSP architecture has self-optimizing and self-healing characteristic making it autonomic. PSP framework is limited to two types of workload shifts i.e. OLTP to DSS and DSS to OLTP shift detection. The framework is also limited to scheduled tasks and does not have ability to manage drastic workload change.

Thereska et. al. [12-15] developed a test bed ‘Ursa Minor’ that is used to predict the workload and provide a direction towards the self-managing system. Ursa Minor has two major components Observer and Stardust and are based on what-if model [16]. The research contributes by identifying and handling show-stopper by enhancing existing mathematical models in shared and distributed systems. They replaces the performance counter with end to end tracing which provides right measurements such as per client, per resource demand and request in distributed and shared systems. They developed a test bed URSA MINOR, a cluster based storage system. It was designed in such a manner that it can easily incorporates in existing as well as new system. They have also devised a modeling infrastructure Observer which is expectation based model to perform predictions in common region of operations. Stardust is the other infrastructure they have developed for the shared, distributed systems. It monitors the service center and critical path of requests while ignoring background and maintenance activities. Ursa Minor [12-15] uses object-based storage which exposes more information about the stored data. The Ursa Minor provides scalability using cluster based technique and dynamic adaptive behavior through online choice. The re-encode process of Ursa Minor takes some extra time of system but throughput increases up to 3 times.

There have been developed numbers of tools techniques and models w.r.t prediction. Existing techniques have number of limitations which have not been addressed. These are the research issues that need to be resolved. In DBMSs, we are interested to self-predict the performance of queries for a given workload. Performance of queries depends on different metrics that is the measure of a database activities and performance. The issues that will be addressed in this research will be that before executing the query how we can manage the workload by deciding, should we run the query or delay and when should it run, how much time it has to wait or if it is problematic query we should kill it. What will be the performance of queries after their execution. There is system sizing problem such as how many CPUs or disks are required for the executing the workload with some time constraint and how much network bandwidth will be needed, capacity planning problem is another issue how can we know that should there is need up gradation or down gradation of system.

III. Proposed Solution

Workload management plays an important role in database management system as well as in data warehouses. Workload management includes many factors if these are predicted in advance then the performance can be enhanced and best optimal results can be achieved. We have proposed a framework for workload management w.r.t prediction- i.e. Workload Predictor. The predictor framework will predict different performance metric such as CPU time, query execution time, etc. There is a need to make the systems, which have self-predicting behavior that will help to manage the workload dynamically and proactively. By doing this the workload will be executed without any delay with maximum efficiency. The Framework as shown in Fig. 1 will predict query run time under load and simultaneously predict the multiple performance metrics including (CPU time, execution time etc).

Objective of the research is to predict the performance metrics (Workload Size, Elapsed time, records accessed, record used, disk I/Os, memory required, message count and bytes) for queries in a given workload. Fig. 2 represents the processing steps of a query. We will take the query execution plan that is generated by the query optimizer. Steps of the proposed predictor framework methodology are as follows:-

A Performance Features Prediction steps:-

1) Building Training Data Model
   a) Training data (query and performance parameters) will be stored in Case Base Reasoning (CBR) and built by extracting query parameters from QEP and performance parameters after their execution.
   b) Actual or test workload is then executed and performance feature vector prediction is performed.
   c) The training data will be divided into different clusters.

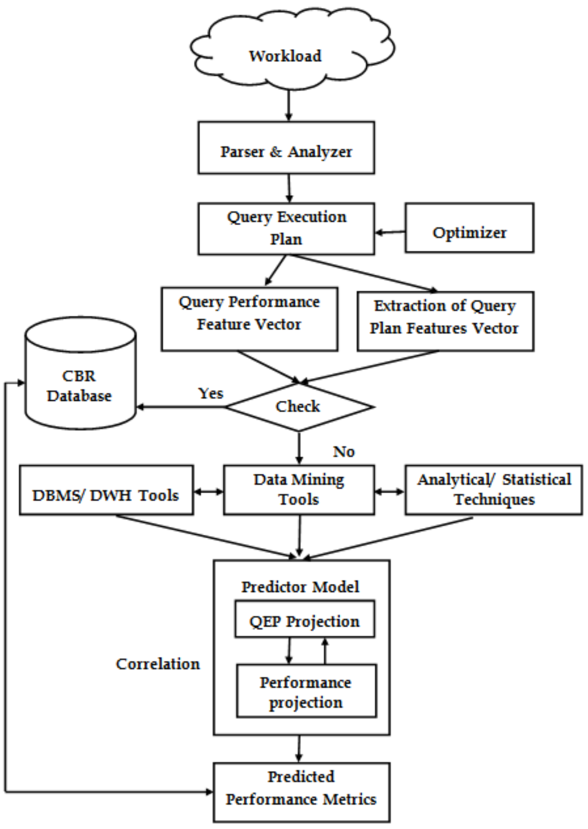

Fig. 1. Predictor framework methodology

2) When a query is entered, it will be parsed and QEP is generated by the optimizer before query execution as shown in Fig. 2. Query feature parameters will be extracted from each QEP that is generated by the query optimizer. These parameters consist of nested queries, selection predicates, join predicates, sort columns and aggregation columns.
3) By extracting the parameters that affect the performance of query, which are workload size, elapsed time, records accessed, record used, disk Input/Outputs, memory, message bytes. Initially these parameters will be extracted and stored by running different queries.
4) For new workload correlation and projection of the query features and performance feature is performed. After query projection distance will be calculated from the nearest cluster, this approach will minimize the searching time.
5) At the end we will get the predicted performance metrics for a query which enables us for decision making for the workload management, capacity planning and system sizing and other related issues.

The contribution of the research is to study and analyze the existing prediction models for DBMS and Data Warehouse (DWH) workload and their performance. Implementation of Kernel Canonical Correlation Analysis Model proposed by Ganapathi et. al. [10] and then enhancing the KCCA model, by improving the performance in the DBMS. We will improve the performance of query by introducing three other parameters (workload size, memory, communication cost) which have impact on performance of the query. Finally we will develop a new workload model based on AI techniques such as CBR and implementing the proposed workload performance prediction framework.

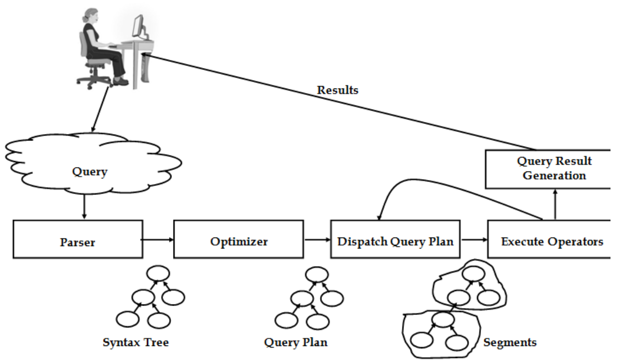

Fig. 2. Query Processing Steps

IV. Conclusion and Future Work

We have proposed a framework for prediction of the performance metrics of DBMS workload. Our predicted metrics will be helpful for decision making about the execution or suspension of the workload. We can also suspend workload for the time being for the queries which are capturing or demanding more resourcing and we do not have at that time. These metrics will also be helpful for scheduling before execution of the query. Our future work is to develop and deploy the test-bed. We will perform various experiments in the presence of different workloads. We will use the performance metrics used by the previous researchers and then other proposed parameters and their effect on the performance of the query. For projection of query feature vector over performance feature we will use the existing technique with some improvements as well as we will apply some other technique to get best results.

Acknowledgment

We wish to thanks the Higher Education Commission (HEC) of Pakistan who is supporting this research work and its implementation.

References

[1] Practical Autonomic Computing: Roadmap to Self Managing Technology, An IBM Journal Paper, 2006.

[2] M. Parashar and S. Hariri, “Autonomic Computing: An Overview,” *Springer-Verlag Berlin Heidelberg*, LNCS 3566, 2005, pp. 247–259.

[3] Jana Koehler, Chris Giblin, Dieter Gantenbein, and Rainer Hauser, “On Autonomic Computing Architectures,” *IBM Zurich Research Laboratory, Switzerland*.

[4] S. R. White, J. E. Hanson, I. Whalley, D. M. Chess, and J. O. Kephart, “An Architectural Approach to Autonomic Computing,” In *the Proceedings of the IEEE International Conference on Autonomic Computing* (ICAC’04), 2004.

[5] Y. Diao, J. L. Hellerstein, S. Parekh, R. Griffith, G. Kaiser, and D. Phung, “Self-managing systems: A control theory foundation,” *12th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*, 2005, pp. 441–448.

[6] Sam S. Lightstone, Guy Lohman, and Danny Zilio, “Toward Autonomic Computing with DB2 Universal Database,” *SIGMOD*, Vol. 31, no. 3, 2002.

[7] A. Mehta., C. Gupta, S. Wang, and U. Dayal, “Automatic Workload Management for Enterprise Data Warehouses,” *IEEE Data Engg*. 31, no.1, 2008, pp. 11-19.

[8] C. Gupta and A. Mehta., “PQR: Predicting Query Execution Times for Autonomous Workload Management,” *International Conferencce on Autonomic Computing*, 2008.

[9] D. Umeshwar, K. Harumi, W. Janet, W. Kevin, G. Archana, and K. Stefan, “Managing operational business intelligence workloads,” *ACM* vol. 43, no. 1, 2009, pp. 92-98.

[10] G. Archana, Harumi A. Kuno, D. Umeshwar, W. Janet, F. Armando, J. Michael, and P. David, “Predicting Multiple Metrics for Queries: Better Decisions Enabled by Machine Learning,” 2009, pp. 592-603.

[11] S. Elnaffar and P. Martin, “An intelligent framework for predicting shifts in the workloads of autonomic database management systems,” *In Proceedings of IEEE International Conference on Advances in Intelligent Systems – Theory and Applications*, 2004.

[12] Abd-El-Malek, M., Courtright II, W. V., Cranor, C., Ganger, G. R., Hendricks, J., Klosterman, A. J., Mesnier, M., Prasad, M., Salmon, B., Sambasivan, R. R., Sinnamohideen, S., Strunk, J. D., and Thereska, E., “Ursa Minor: versatile cluster-based storage,” *In Conference on File and Storage Technologies*, pp. 59–72, 2005.

[13] E. Thereska, D. Narayanan, A. Ailamaki, and G. R. Ganger, “Observer: keeping system models from becoming obsolete,” *Workshop on hot topics in autonomic computing*, 2007.

[14] Thereska, E., Narayanan, D., and Ganger, G. R., “Towards self predicting systems: What if you could ask “what-if”?”, In *3rd International workshop on self-adaptive and autonomic computing systems*, 2005.

[15] D. Narayanan, E. Thereska, and A. Ailamaki, “Continuous resource monitoring for self-predicting DBMS,” In *International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems* (MASCOTS’05), 2005, pp. 239–248.

[16] E. Thereska, B. Salmon, J. Strunk, M. Wachs, M. Abd-El- Malek, J. Lopez, and G. R. Ganger. Stardust, “Tracking activity in a distributed storage system,” In *ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, 2006, pp. 3–14.

[17] M. Mehta and D. J. DeWitt, “Dynamic Memory Allocation for Multiple-Query Workload,” In *Proc. of the Nineteenth International Conference on Very Large Data Bases (VLDB)*, 1993.

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

[19] A. M ̈onkeberg and G. Weikum. “Performance evaluation of an adaptive and robust load control method for the avoidance of data-contention thrashing,” *VLDB*, 1992.

[20] M. J. Carey, M. Livny, and H. Lu, “Dynamic Task Allocation In A Distributed Database System,” In *Proc. of the 5th International Conference on Distributed Computing Systems (ICDCS)*, 1985, pp. 282–291.

[21] S. Elnaffar, P. Martin, and R. Horman, “Automatically classifying database workloads,” 2002. [Online]. citeseer.ist.psu.edu/ elnaffar02automatically.html.

[22] S. Elnaffar, “A Methodology for Auto-Recognizing DBMS Workloads,” *Proceedings of Centre for Advanced Studies Conference* (CASCON ’02), 2002, pp1-15.