IJCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 01, May 2011
ISSN (Online): 2231 –5268
www.ijcsms.com

Enhanced Performance of Database by Automated Self-Tuned Systems

Ankit Verma
Department of Computer Science & Engineering, I.T.M. University, Gurgaon (122017)
ankit.verma.aquarius@gmail.com

Abstract
Performance tuning of Database Management Systems (DBMS) is complex as well as challenging task since it involves identification and alteration of several key performance tuning parameters. The quality of tuning and the extent of performance enhancement achieved greatly depend on the skill and experience of the Database Administrator (DBA). The ability of our automated database design to adapt to dynamically changing inputs makes them ideal candidates for employing them for tuning purpose. In this paper, a novel tuning algorithm based on new script estimated tuning parameters is presented. The key performance indicators are proactively monitored and fed as input to the proposed script and the trained networks estimates the suitable size of the buffer cache, shared pool and redo log buffer size. The tuner alters these tuning parameters using the estimated values using a rate change computing algorithm. The preliminary results show that the proposed method is effective in improving the query response time for a variety of workload types. To summarize, this paper presents a self tuned database system or we can say, automated database system whose main focus is performance optimization.

Keywords: *Self-tuned database, automated database, database performance, database tuning, DBA, Buffer Miss Ratio, Data Miner, Buffer Cache.*

I. INTRODUCTION

DATABASE Management Systems (DBMSs) can be defined as the systems which manage the large databases as shown in fig 1. DBMS is a suite of services (software applications) for managing databases, which enables simple access to data, allows multiple users access to the information and manipulates the data found in the database.
Database Management Systems have a wide range of application in real life such as in any corporate house, the online systems, and e-commerce applications. Thus, to provide reliable services with quick query response times to the customers, the Database Management Systems (DBMS) must function efficiently and should have built-in support for quick system recovery time in case of partial failure or system resource bottlenecks. The performance of these systems is greatly influenced by several factors which include database size which keeps on growing with its usage over a period of time, increased user base, sudden increase in the user processes, improperly or un-tuned DBMS. All these factors degrade the system response time and hence a system is required that would anticipate the performance Degradation by carefully monitoring the System performance indicators and would auto tune the system. In other words, a self-tuned database system is desired for the optimization of query response time.

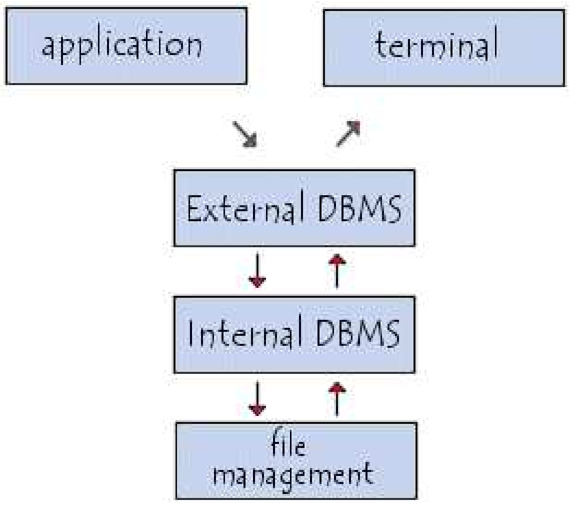


Fig.1: Database Management System

Many commercial applications require huge databases for the purpose of data aggregation as well as data distribution. As there are limited storage resources available performance enhancing features such as selection of indexes, materialized views, horizontal and vertical partitions is a challenging optimization problem. Thus, the task of data aggregation as well as data distribution can be made somewhat automatic so that it may incorporate dynamic changes effectively.
Database maintenance, being a continuous process involves considerable effort on the part of a Database Administrator (DBA) for a particular enterprise. The DBA has to monitor several system parameters and fine tune them to keep the system functioning smoothly in the event of reduced performance or partial failure. This parameter monitoring and maintenance is a complex and time consuming task. It is therefore desirable to build a system that can tune itself and relieve the DBA from this tedious and error prone task of tuning. Although Oracle 9i and 10g have built in support for tuning in the form of tuning advisor. The advisor estimates the optimal values of the tuning parameters and recommends them to the

IJCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 01, May 2011
ISSN (Online): 2231 –5268
www.ijcsms.com

DBA. A similar advisor is also available in SQL Server 2005 which is based on what-if analysis. In this approach, the DBA provides a physical design as input and the tuning advisor performs the analysis without actually materializing the physical design. However, the advisor available in 2005 recommends the changes needed at the physical level such as creation of index on tables or views, restructuring of tables, creation of clustered index etc. which are considered to be very expensive in terms of Database Server down time and the effort on part of the DBA.

RELATED WORK

Database Administrator is responsible for enhancing the performance of database system. The detection of performance degradation is achieved by continuously monitoring system performance parameters. Several methods including the usage of materialized views and indexes, pruning table and column sets [1-2], usage of self healing techniques [3-4], usage of physical design tuning etc have been proposed that proactively monitor the system performance indicators, analyze the symptoms and auto tune the DBMS to deliver enhanced performance. The performance degradation is due to increased workload on the system. This increased load has to be minimized to enhance the response rate of the system. In order to achieve this objective, either the administrator decreases some amount of load by closing some files or he may increase the RAM as shown in fig 2. The administrator has to check continuously or we can say, at regular intervals the Buffer Cache Hit (BCH) ratio. Based on this hit ratio, the database administrator determines if more amount of RAM has to be allocated. This task of load reduction by increasing RAM requires manual intervention and thus may take even years to complete.

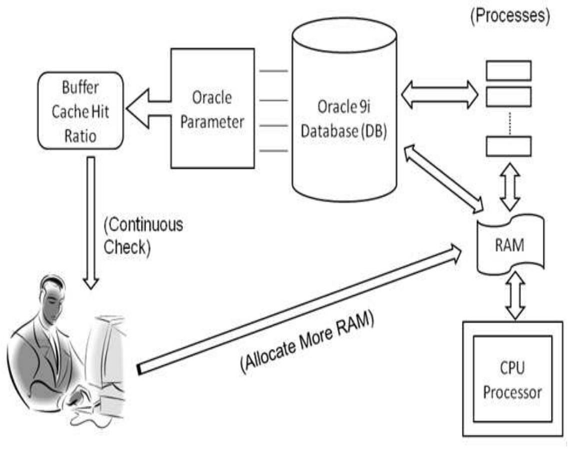


Fig.2: Manual Database Design

However, Oracle manages RAM memory demands according to the demands of each task by using sophisticated algorithms to improve the speed of RAM intensive tasks. Oracle DBA can dynamically de-allocate RAM memory as well as re-allocate it. But since database administrator is a normal human being, he cannot calculate the actual amount of RAM memory required by an application. Due to this limitation of DBA, the allocation of RAM manually for optimizing performance of database system becomes a complicated as well as costly task. Sometimes, more amount of RAM is allocated than needed which wastes the extra portion of RAM. Thus, there is a great need of dynamic memory allocation features to create a self tuning database. In Oracle Database 10g, a self tuning feature such as Automatic Memory Management (AMM) allows the database system to detect shortages and adjusts the main memory regions according to the changing demands on the Oracle environment. Therefore, researchers are now focusing on the development of self tuning techniques such as the COMFORT automatic tuning project [5] or the MAPE approach given by IBM [6] for a continuous adaptation. Ranking of various tuning parameters based on statistical analysis is presented in[7]. The ranking of parameters is based on the amount of impact they produce on the system performance for a given workload. A formal knowledge framework for self tuning database system is presented in[8] that defines several knowledge components which include Policy knowledge, Workload knowledge, Problem diagnosis knowledge, Problem Resolution Knowledge, Effectors knowledge, and Dependency knowledge. The architecture presented in this paper involves extracting useful information from the system log and also from the DBMS using system related queries. This information gathered over a period of time is then used to run the SQL scripting for a desired output response time. The application framework would then estimate the extent of correction to be applied to the key system parameters that help scale up the system performance. The classical control is modified and a three stage control involving Monitor, Analyze and Tune [7] is employed to ensure system stability. The architecture presented in [9] for self healing database forms the basis for the new architecture presented in this paper. This paper presents a new DBMS architecture based on modular approach, where in each functional module can be monitored by set of monitoring hooks. These monitoring hooks are responsible for saving the current status information or a snapshot of the server to the log. This architecture has high monitoring overhead, due to the fact that when large number of parameters to be monitored, almost every module's status information has to be stored on to the log and if done frequently may eat up a lot of CPU time.

Moreover, this architecture focuses more on healing the system and does not consider tuning the DBMS for performance improvement.

IJCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 01, May 2011
ISSN (Online): 2231 –5268
www.ijcsms.com

PERFORMANCE TUNING

Calibrating the system for desired response time is called performance tuning. The objective of this system is to analyze the DBMS system log file and apply information extraction techniques and also gather key system parameters like buffer miss ratio, number of active processes and the tables that are showing signs of rapid growth. The control architecture presented in this paper, only one parameter namely, the buffer cache is tuned. Using the statistical information of these three parameters to train the Neural Network and generate an output that gives an estimate of the optimal system buffer size. Since, the DBMS are dynamic and continuously running around the clock, the above information must be extracted without causing any significant system overhead. Extracting information from system log ensures that there is no overhead. The queries that are used to estimate buffer miss ratio, table size and number of user processes are carefully timed and their frequency is adjusted so that it does not add to the overhead in monitoring the system.

PROPOSED ARCHITECTURE

Many business applications demand the use of complex database systems which should be administered and optimized for better performance. As suggested in [2], physical tuning should be avoided as it is expensive. As the physical design of database suffers from various limitations, a new script based automated database architecture is proposed in order to achieve high grade of performance. The architecture as shown in fig 3 is employed for identifying the symptoms and altering key system parameters. The DBMS system log file will be the primary source of information checks the current status of the system. The data miner tool compresses the data into smaller information base since the log file may contain huge amount of data. The architecture has three basic building blocks comprising of Data Miner, Script and Tuner. After the extraction of meaningful information, the extent of correction required is estimated by the proposed script and algorithms

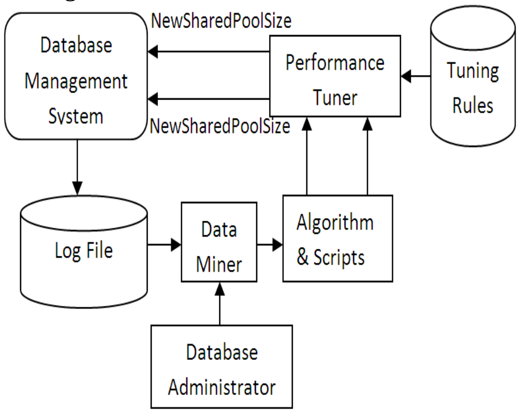


Fig 3. Script based Tuning Architecture

These algorithms and scripts would tune the database using various tuning rules as well as system parameters. However, several parameters can be altered simultaneously for better performance gain. The algorithm estimates the required buffer size based on the current DBMS input parameters and the tuner applies the necessary correction to the buffer size based on the tuning rules. Most importantly the internal corrective measure such as altering the buffer size of the DBMS used in query processing is explored in this architecture.

Proposed Approach

In this research proposal, we would provide a self tuned database system as shown in fig 4 in order to enhance system performance. Since DBA is responsible for administration and optimization of various tasks, he can either increase RAM or can decrease the amount of load on CPU for the purpose of performance optimization. But this would be time consuming technique as DBA is a normal human being who cannot perform complex calculations within seconds like a computer system.
DBA may not know exactly how much RAM is to be allocated for enhancing system performance. So, we propose an approach to automate this optimization task of DBA as shown in fig i.e. the task which DBA has to do for performance enhancement would now be done by the computer system within small timelines.

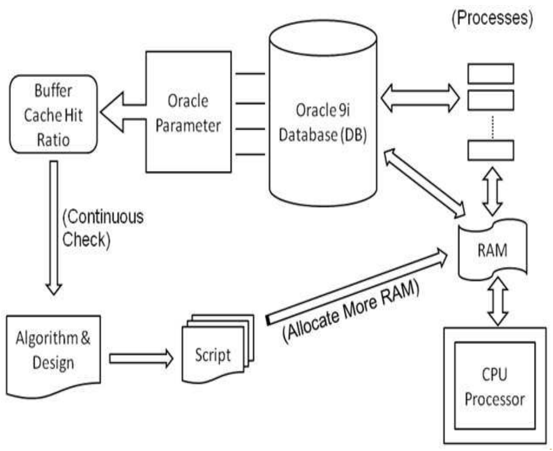
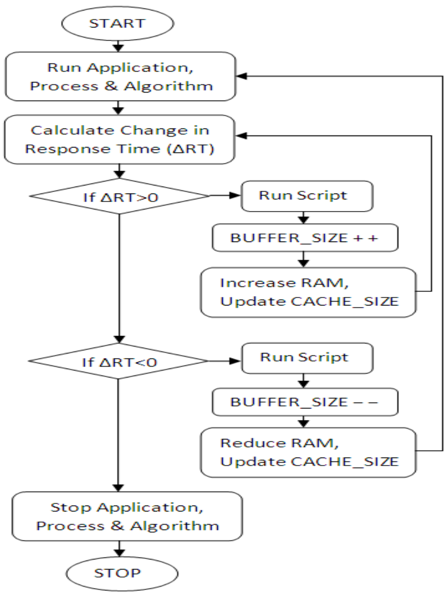


Fig.4: Automated Database Design

IJCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 01, May 2011
ISSN (Online): 2231 –5268
www.ijcsms.com

Algorithmic Flow Chart

The block diagram representation of proposed approach is as under:


START
Run Application, Process & Algorithm
Calculate Change in Response Time (ΔRT)
If ΔRT>0
Run Script
BUFFER_SIZE + +
Increase RAM, Update CACHE_SIZE
If ΔRT<0
Run Script
BUFFER_SIZE – –
Reduce RAM, Update CACHE_SIZE
Stop Application, Process & Algorithm
STOP

Proposed Algorithm

The algorithm defines three variables: _RT abbreviates for change in response time, BUFFER_SIZE denotes the current size of buffer, CACHE_SIZE corresponds to the size of cache memory.

```
ALGORITHM

1. dbTuner (ESTIMATED_CACHE_SIZE)
2. Begin
3. Run application, algorithm and process
4. Calculate the change in response time (_RT)
5. If (_RT>0)
{
Run Script
{
BUFFER_SIZE = BUFFER_SIZE + 1
Allocate more RAM and update CACHE_SIZE
}
Else IF (_RT<0)
{
Run Script
{
BUFFER_SIZE = BUFFER_SIZE - 1
Reduce RAM and update CACHE_SIZE
}
}
6. Stop application, algorithm and process
7. End
```

EXPERIMENTAL RESULTS

UICCs Table I shows the sample training data. A training data set of size 100 was used to test the proposed system. As can be seen from the table, the buffer size is adjusted for increased table size, Number of user processes and Buffer Miss Ratio so that query execution time is reduced and the memory is used efficiently.

Tab. Size (in no. of records)	Buff.Miss Ratio	Shared Pool size (in MB)	Buff. Size (in MB)
1000	0.9624	32	4
1000	0.9152	32	4
1000	0.9791	32	8
1000	0.9613	32	8
2000	0.9371	32	8
2000	0.9453	40	8
3000	0.8931	40	16
3000	0.8253	40	16

Table I. Sample Training Data Set

The experiment was carried on Oracle 9i with 100 internal nodes. The estimated buffer size generated by the script is based on the dynamic values of the above three parameters as input.
The tuner takes this input and alters the buffer size. The results obtained are really promising. As can be seen from the output in Fig. 4 the execution time is significantly lower for the increasing value of the buffer size. The query used takes join of three tables and generate huge dataset as result.

CONCLUSION

For the purpose of optimizing performance in data base systems, a new automated physical design is proposed. An improvement in the physical design automatically has become the top research field both in academics and industry. In this paper, we proposed a typical automated physical design and two techniques for improving its efficiency and performance.

REFERENCES

[1] S. Agarwal and et.al., “Automated selection of materialized views and indexes”, VLDB, 2007.

[2] Surjit Choudhuri, Vivek Narasayya, “Self tuning database systems : A Decade progress”, Microsoft Research. 2007.

[3] Philip Koopman, “Elements of the Self-Healing System Problem Space”, IEEE Data Engineering Bulletin. 2004.

IJCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 01, May 2011
ISSN (Online): 2231 –5268
www.ijcsms.com

[4] Peng Liu, "Design and Implementation of Self healing Database system",IEEE Conference, 2005.

[5] Sattler, K, Geist, I. Schallehn, E. Quiet: Continuous query-driven index tuning. In Proceedings of VLDB, 2003.

[6] Schnaitter, K., Abiteboul, S., Milo, T., Polyzotis, N.: COLT: continuous on-line tuning. In Proceedings of ACM SIGMOD Conference 2006.

[7] Debnath, B.K.; Lilja, D.J.; Mokbel, M.F., "SARD: A statistical approach for ranking database tuning parameters", Data Engineering Workshop, 2008. ICDEW 2008. IEEE 24th International Conference, April 2008.

[8] Bruno N. and Chaudhuri S. Physical Design Refinement. The Merge-Reduce Approach. In Proceedings of the EDBT Conference, 2006.

[9] Rimma V. Nehme, "Dtabase, Heal Thyself", Data Engg. Workshop April 2008.