Architecture of Automated Database Tuning Using SGA Parameters

Hitesh KUMAR SHARMA[1], Aditya SHASTRI[2], Ranjit BISWAS[3]
[1]Assistant Professor, ITM University
[2]Vice-Chancellor, Banasthali University, Rajasthan-304022, India
[3]Head and Professor,CSE Dept, Jamia Hamdard (Hamdard University)
hiteshsharma@itmindia.edu, adityashastri@yahoo.com, ranjitbiswas@yahoo.com

Business Data always growth from kilo byte, mega byte, giga byte, tera byte, peta byte, and so far. There is no way to avoid this increasing rate of data till business still running. Because of this issue, database tuning be critical part of a information system. Tuning a database in a cost-effective manner is a growing challenge. The total cost of ownership (TCO) of information technology needs to be significantly reduced by minimizing people costs. In fact, mistakes in operations and administration of information systems are the single most reasons for system outage and unacceptable performance [3]. One way of addressing the challenge of total cost of ownership is by making information systems more self-managing. A particularly difficult piece of the ambitious vision of making database systems self-managing is the automation of database performance tuning. In this paper, we will explain the progress made thus far on this important problem. Specifically, we will propose the architecture and Algorithm for this problem.
Key Words: *Database Tuning, DBA, SGA, SGA Parameters, Automated Tuning, TOC.*

1 Introduction

As we have seen, hardware costs fall rapidly while human costs remain relatively static. This leads to a condition there the human costs of manual tuning activities outpaces the costs of faster hardware (see figure 1). Most large databases are managed by DBAs who are responsible for the good performance of the database but manual physical design is both time consuming and very tedious, as the database administrator (DBA) needs to find the benefits of different individual design features that can possible interact with one another. Motivated not only by the difficulty of tuning but also from the need to reduce the total cost of ownership in their products, several commercial DBMS vendors offer automated physical design tools with several features.

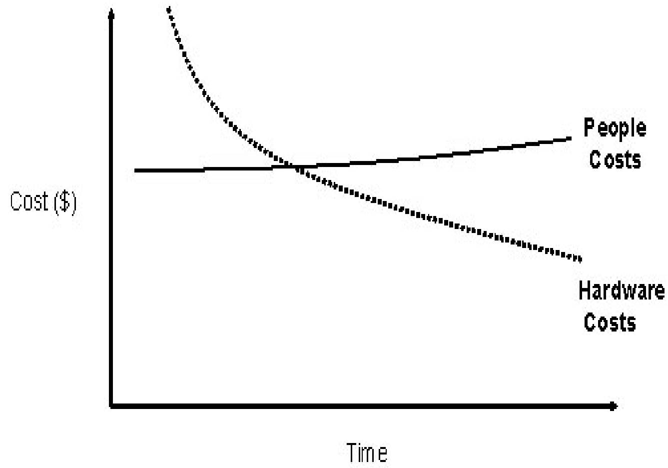

Fig.1. H/W cost vs. Human Cost [Ref. 2]

With the dramatic drop of hardware and software prices, the expenses due to human administration and tuning staff dominate the cost of ownership for a database system [4].

The physical design problem involves searching a potentially very large space of different candidate configurations. Searching the space of alternative configurations is impractical. Therefore, recent physical design tools are based on greedy heuristics that prune the search space. The repeated calls to the optimizer each time we want to evaluate a query under a different configuration impose a serious bottleneck in the execution of physical designers. Based on experimental results 90% of the tuning time is spent on waiting results from the optimizer instead of evaluating potentially promising configurations [5,6].

2. Performance Tuning

Most systems will respond to increased load with some degree of decreasing performance. A system's ability to accept higher load is called scalability, and modifying a system to handle a higher load is synonymous to performance tuning.
Systematic tuning follows these steps:

- Assess the problem and establish numeric values that categorize acceptable behavior.
- Measure the performance of the system before modification.
- Identify the part of the system that is critical for improving the performance. This is called the bottleneck.
- Modify that part of the system to remove the bottleneck.

A performance problem may be identified by slow or unresponsive systems. This usually occurs because high system loading, causing some part of the system to reach a limit in its ability to respond. This limit within the system is referred to as a bottleneck. A handful of techniques are used to improve performance.
Data drives today's businesses, and managing databases often involves complex planning, time management and system wide routine task implementation. Database automation helps enterprises better manage their database operations, reducing down-times as well as the overall time taken in database management. Automation anywhere works with any SQL database, like Oracle, MS SQL, Sybase, SQL DB2, etc. Unlike other automation solutions, it does not require significant training. Simple, easy-to-use yet powerful, it can automate any database task.[1]

3. Manual System Architecture

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, usage of self healing techniques, 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. 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.[1]
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.

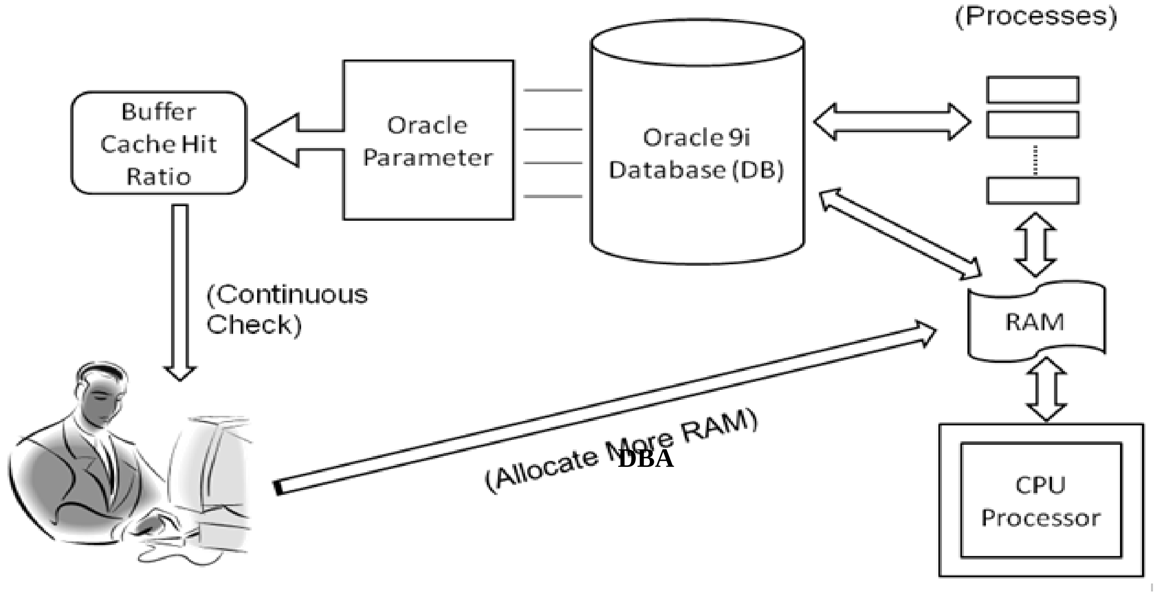

Fig.2. Manual Database Design

Sometimes, more amount of RAM is allocated than needed which wastes the extra portion of RAM. [3] 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 that define 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 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.

4. Automated System 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 figure 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.

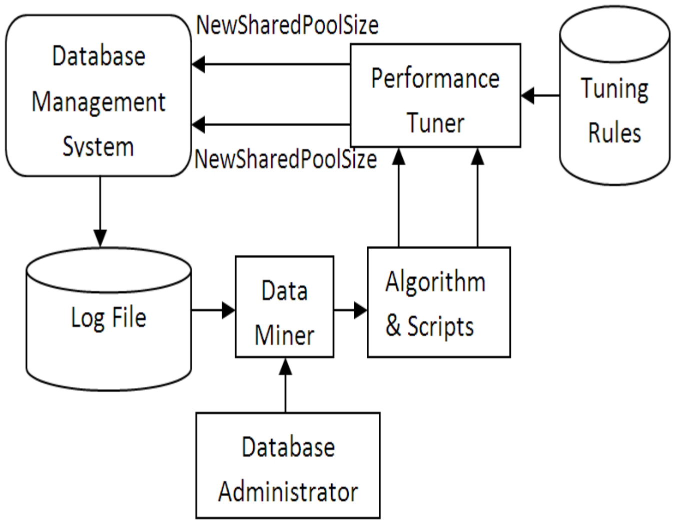

Fig.3. Script based Tuning architecture

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.

In this research proposal, we provided a self tuned database system architecture 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. [5]

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.

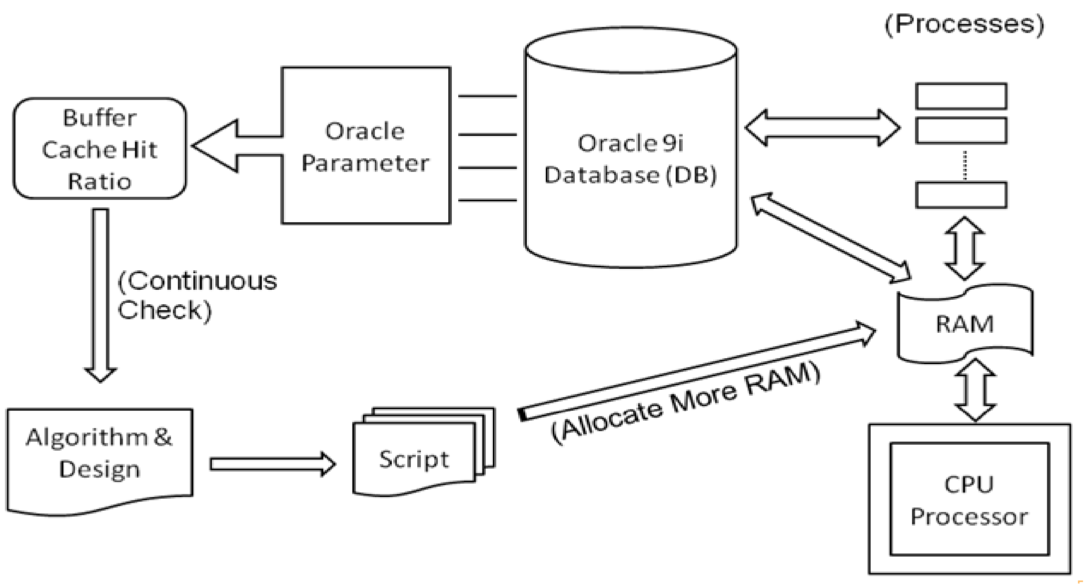

Fig.4. Automated Database Design

5. Algorithm & Flow Chart

The algorithm defines three variables: ΔRT 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 (ΔRT)
5.                If (ΔRT>0)
                  {
                          Run Script
                          {
                                  BUFFER_SIZE = BUFFER_SIZE + 1
         Allocate more RAM and update CACHE_SIZE
         }
      Else IF (ΔRT<0)
      {
       Run Script
       {
                BUFFER_SIZE = BUFFER_SIZE - 1
         Reduce RAM and update CACHE_SIZE
         }
       }
 6.   Go To Step 4
 6.   Stop application, algorithm and process
 7.  End
```

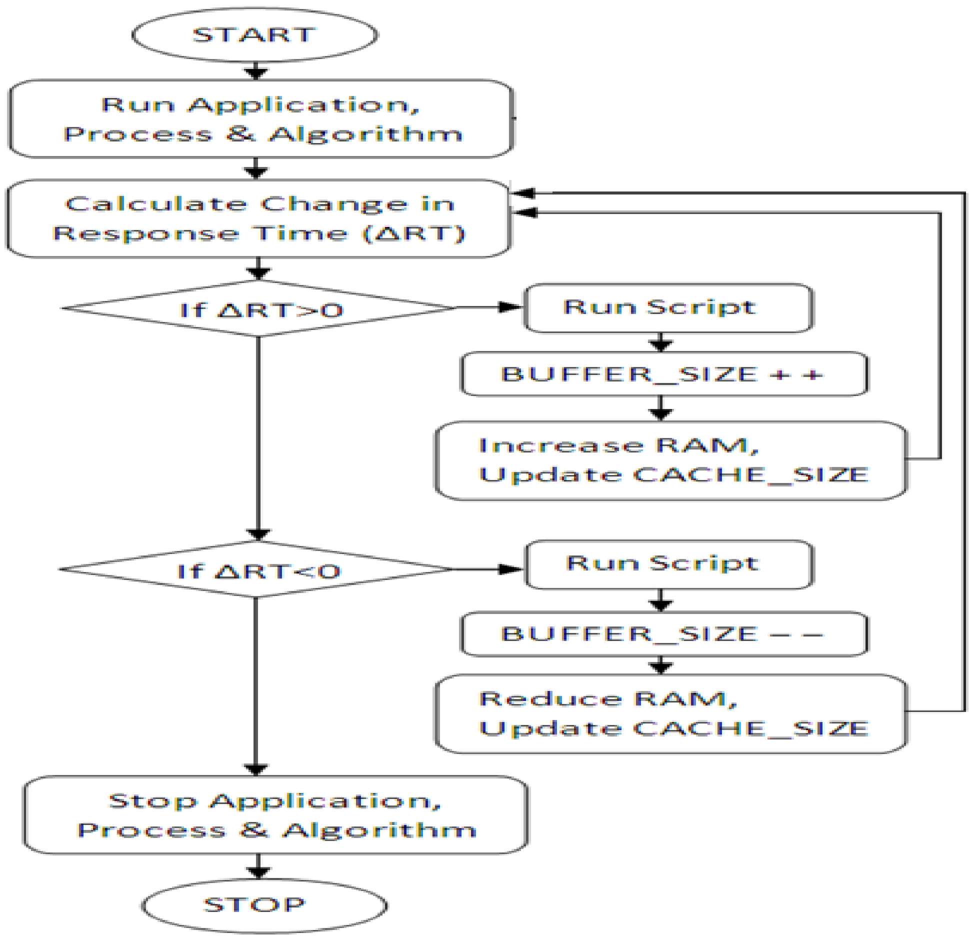

Fig.5. Flowchart for Automated Tuning

6. Experimental Results

Table 1 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.

Table 1. Sample Training Data Set

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

7. Conclusion

Tuning the database can become quite complex, but Oracle9i offers the administrator an unparalleled ability to control the PGA and SGA. Until Oracle9i evolves into a completely self-tuning architecture, the DBA will be responsible for adjusting the dynamic configuration of the system RAM. Automated SGA adjustment scripts can be used to allow theDBA to grow and shrink the SGA regions. These scripts are placed in dbms_job for scheduled processing. Oracle provides enhanced views in v$process, v$pgastat to allow you to monitor the behavior of the RAM sort area The v$ views in Oracle9i also provides insights about the RAM usage for individual SQL statements within the library cache.

References

[1] I. Alagiannis, *Towards Adaptive, Flexible, and Self-tuned Database Systems*, (DIAS, I&C, EPFL) in EDIC-ru/05.05.2009

[2] D. Burleson, *Oracle Tuning,* The Definitive Reference Second Edition.

[3] *Foundations of Automated Database Tuning*, VLDB '06, September 12–15, 2006, Seoul, Korea.Copyright 2006 VLDB Endowment, ACM

[4] *Rethinking Database System Architecture: Towards a Self-tuning RISC-style Database System in Cairo*, Egypt, 2000

[5] *AutoAdmin:Self-Tuning Database Systems Technology*, Copyright 2006 IEEE

[6] *Self-Tuning Database Systems: A Decade of Progress*, Copyright

[7] *SQL Memory Management in Oracle9i*, Hong Kong, China, 2002

Hitesh KUMAR SHARMA,
The author is An Assistant Professor in ITM University. He has published 8 research papers in National Journals and 2 research paper in International Journal. Currently He is pursuing his Ph.D. in the area of database tuning**.**

Aditya SHASTRI,
Ph.D. MIT, Published about 200 research papers in international journals on Graph Theory with applications in Communication, Computer Graphics and Parallel Processing,Vice Chancellor, Director, Banasthali University, Banasthali, INDIA

Ranjit BISWAS,
Associate Director, MIRU, Faridabad, Published about 100 research papers in International journals/bulletins of USA/Europe, out of which more than 40 papers are independently published papers and the rest are published jointly with other authors (with Ph.D. scholars). Haryana, INDIA