Fuzzy Controlled Architecture for Performance Tuning of Database Management System

S. F. Rodd
Gogte Institute of Technology
Udyambag,
Belgaum-Karnataka, INDIA

Umakant P. Kulkarni
SDM College of Engineering & Technology
Dharwar, Karnatka, INDIA

A. R. Yardi
Walchand College of Engineering
Maharashtra, INDIA

ABSTRACT
Database Management Systems deliver higher performance only when they are properly tuned. Database tuning is complicated due to the fact that several conflicting tuning parameters have to be adjusted simultaneously for a variety of workload types and the highly unpredictable traffic patterns. The Database Administrator(DBA) has to be an expert and using his experience and expertise must judiciously decide the extent of tuning of the most important tuning factors so as to ensure the required level of performance in terms of response time and throughput. The process of tuning being complex and to keep the cost of ownership low, it is desirable to build self tuning database systems. In this paper, a new tuning architecture based on fuzzy logic is presented, where in, the control action is expressed in linguistic terms. In this system the key performance indicators are fuzzified, appropriate fuzzy rules are employed to estimate the extent of tuning required for a few important tuning parameters. After defuzzification, a control action is initiated to scale up the system performance. The experimental results obtained for different workload types and the user load, indicate that it is possible to significantly improve the query response time using this technique.

General Terms
Performance Tuning, Fuzzy Logic, Database Management Systems.

Keywords
Buffer Hit Ratio, Workload, Fuzzy Rules, Tuning, Database Administrator.

1. INTRODUCTION
Database Management Systems are an important component of any business enterprise. The applications that solve business problems are rarely built with optimal performance in mind. With the prolonged usage of the applications, database size, number of users, and the workloads grow dramatically. A badly tuned database may not be able to handle this changed computing ambience, resulting in very poor query response time and as a consequence reduced throughput. An ill-tuned system not only performs badly, but also results in poor usage of system resources such as CPU, Memory, and Disk I/O etc. The role of the DBA is to keep track of the key performance indicators such as Buffer Hit Ratio(BHR), Query Response Time and also overall database size and initiate appropriate measures so to scale up the system performance. It is rare to find expert DBAs and expensive to employ them, thus resulting in higher cost of ownership. It is therefore desirable to have a self tuning system, where in the system itself keeps track of the performance indicators, employ some kind of an intelligent decision making mechanism and initiate a process that quickly scales up the system performance in terms of either the query response time or throughput. Most modern database systems provide facilities to dynamically alter several of their tuning parameters at run time. This feature and coupled with knowledge of effect of tuning several tuning parameters, it is now possible to build self tuning systems. Several such self tuning systems have been proposed in literature to auto tune the DBMS for significant performance gains.

The problem of tuning a database can be treated as a conventional control system. However, the query processing inside the DBMS is so very complex, that a DBMS can't be modeled using the conventional control theory. Hence it is desirable to employ those techniques that are best suited to handle such complex systems. These include, use of Neural Networks, Fuzzy Logic and Genetic Algorithms or a combination of these soft computing techniques. In this paper, an attempt has been made to use Fuzzy logic to perform tuning of the DBMS. Use of Neural network to fine tune the DBMS has already been established. As an enhancement, a Neuro-Fuzzy system can also be a choice to make the system learn from past experience and intelligently alter the most effective tuning parameters in their useful range and keep the DBMS performing at its peak performance level at all times.

2. RELATED WORK
Self tuning is a concept derived from autonomic computing and self healing systems. Several methods have been proposed that proactively monitor the system performance indicators analyze the symptoms and auto tune the DBMS to deliver enhanced performance. Use of Materialized views and Indexes, Pruning table and column sets[1-2], use of self healing techniques[3-4], use of physical design tuning are among the proposed solutions. The classical control is modified and a three stage control involving Monitor, Analyze and Tune[5] is employed to ensure system stability. The architecture presented in [6] for self healing database 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 consume lot of CPU time. Moreover, this architecture focuses more on tuning the DBMS for performance improvement under a variety of load conditions. Application of fuzzy logic in control applications is presented

in [7]. Ranking of various tuning parameters based on statistical analysis is presented in [8]. 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[9] that define several knowledge components. The knowledge components include Policy knowledge, Workload knowledge, Problem diagnosis knowledge, Problem Resolution Knowledge, Effecter knowledge, and Dependency knowledge. Database tuning using the virtualization concept is presented in[10] where the system resources like CPU time, I/O Bandwidth and Memory are allocated to the several virtual computing environments with each DBMS running on each of these virtual machines attending to a particular type of workload. [12-17] present the concept of self tuning and the techniques to implement in some of the proprietary database systems like Oracle 10g and DB2.

The architecture presented in this paper, involves extracting useful information from the DBMS using system related queries. Applying fuzzy rules on fuzzy variables using the linguistic terms and determine the extent of correction to be applied to the key system parameters that help scale up the system performance.

3. EFFECT OF TUNING ON RESPONSE TIME

A detailed experimentation was carried to estimate the impact of the two important tuning parameters namely, the buffer cache size and the shared pool size with OLTP(OnLine Transaction Processing) and DSS(Decision Support System) type workloads.

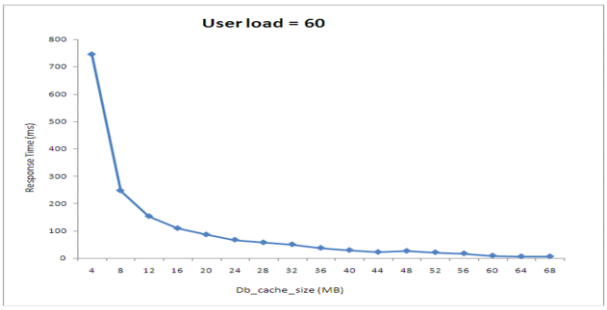

Figure 1. Effect of Buffer cache on the Response time(OLTP)

Figure 1 shows that the buffer caches size has a significant impact on the response time with OLTP type workload and can be appropriately adjusted to boost the response time. Similar shaped graph was observed for shared pool only with a higher value of response time. Figure 2 shows the effect of buffer cache on the response time for a DSS type work load. The response time can again be brought down significantly by altering the buffer cache size for DSS workload type.

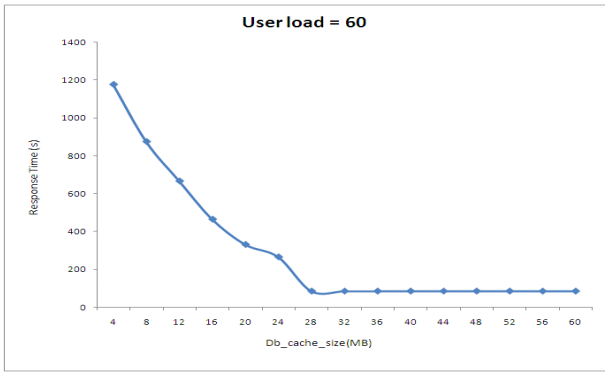

Figure 2. Effect of Buffer cache on Response time(DSS)

4. FUZZY CONTROLLED TUNING

Fuzzy logic is most suitable choice for many control applications for the fact that fuzzy control systems are robust, can be tweaked easily to improve the system performance dramatically and most importantly they are much simpler in design to implement. Moreover, there is no need to measure rate of change of the input parameters and the number of inputs and outputs are not limited to small number.

Calibrating the system for desired response time is called performance tuning. The objective of this system is to analyze the DBMS, by proactively monitoring the performance indicators like buffer miss ratio, number of active processes and the table's size that are showing signs of rapid growth and initiate control measure using fuzzy control. In this paper, a new self tuning architecture based on fuzzy logic is proposed. This architecture comprises of a module that continuously tracks the system performance my noting the important performance indicators. The fuzzyfier module maps these performance indicators to fuzzy variables. The fuzzy control module uses the fuzzy rules comprising of IF THEN kind of statements on the fuzzy input variables to decide on the fuzzy output variables. The fuzzy output variables in this case would be the tuning parameters of the DBMS. Having obtained the extent of tuning required in fuzzy terms, a defuzzyfier module generates the crisp output parameters that are eventually used by the tuner module to fine tune the DBMS. As shown in Fig. 3, two input parameters namely Buffer Hit Ratio(BHR) and number of Active Users(N) are gathered from the DBMS and are used as inputs to the fuzzy control system.

The tuner may tune one parameter at a time or may alter several of them simultaneously as warranted by the dynamic conditions of the system. Fig. 3 shows the details of this control architecture. In the proposed system two output parameters namely, buffer cache size and shared pool size are tuned.

4.1 Fuzzification of Input Parameters

Identifying key performance indicators must be based on a careful characterization study that explores the impact of various tuning parameters on these performance indicators. In this experimental setup, Buffer Hit Ratio and User load are the two parameters that have significant impact on the performance are chosen as input to the fuzzifier module. The membership functions chosen for the input variables are as under

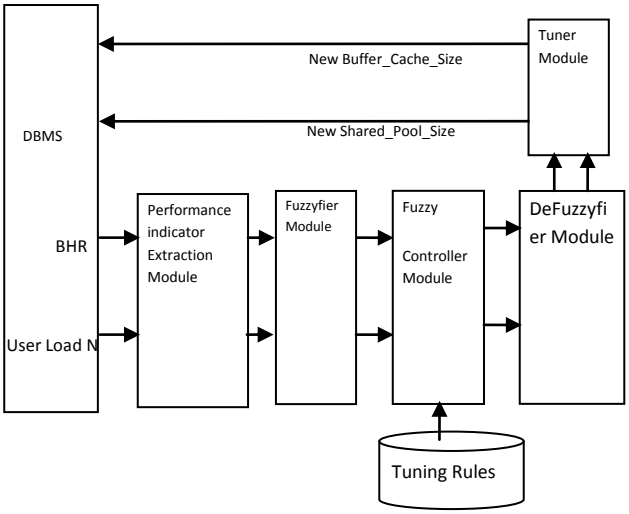

Figure 3. Fuzzy Controlled DBMS Performance Tuning Architecture

TABLE 1. Membership functions for input/ output

Variable name	Membership function
Buffer Hit Ratio	Gaussian
No. of Users	Triangular
Buffer Cache Size	Gaussian

To fuzzify the input, it is divided into four groups of values and membership functions like Gaussian or Triangular are used as shown in Fig. 4. The resulting sets of values are fed to the fuzzy Controller. The fuzzy controller employs the fuzzy rules that are constructed based on the experience and expertise of the database administrator. For example it is imperative that when user load is high and buffer hit ratio is moderate, the buffer cache size must be set to moderately high value. It is an iterative process to arrive at final fuzzy control rules that give the best performance results.

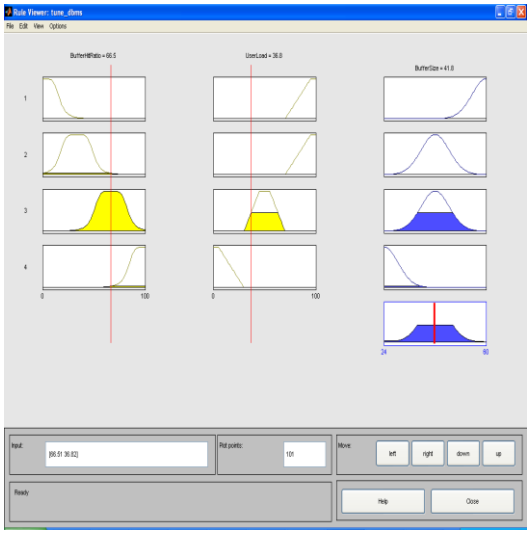

Figure 4. The fuzzy values of BHR, N and Buffer Cache with respective member functions

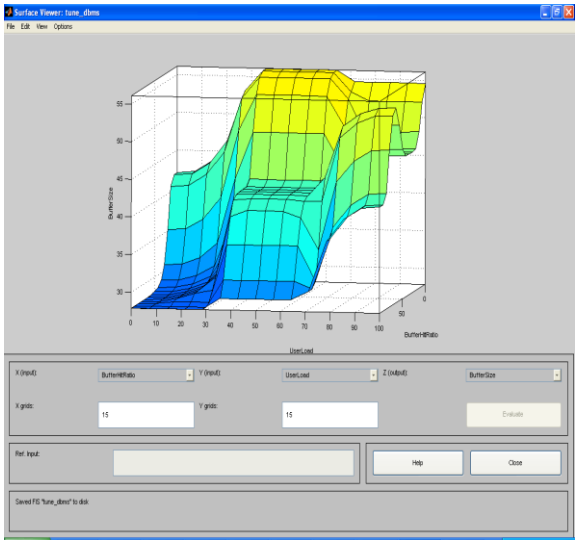

Figure 5. Buffer Cache size as a function of Buffer Hit

4.2 Fuzzy Control Rules

Fuzzy control rules are based on general logical reasoning and based on programming language constructs namely the IF THEN control construct. Linguistic terms are used to describe the extent of tuning required. These rules must be carefully framed and most of the time requires modification till desired

results are obtained. For instance, to find the new buffer caches size, some of the fuzzy rules could be formed as under:

Rule 1:

IF buffer hit ratio is *high* AND number of user load is *low* THEN set buffer cache size to *low.*

Rule 2:

IF buffer hit ratio is *moderate* AND number of user load is *high* THEN set buffer cache to *high.*

Fig. 5. shows the effect of applying the fuzzy rules on the Buffer Cache as a function of Buffer Hit Ratio and Number of users. As the Number of users increase, the buffer caches size also rises, taking into account the value of buffer hit ratio. For instance if the Buffer Hit Ratio is very low and Number of users is very high, the buffer cache size must be extremely large.

4.3 De-fuzzification of Output Parameters and Control Action

Having obtained the value of the tuning parameter in fuzzy terms, defuzzification is to be carried again employing certain member functions to obtain the crisp values for the tuner. The tuner then initiates a control action by dynamically altering the value of the tuning parameter. In this case, it is the buffer cache and the shared_pool_size that are altered and the effect is estimated in terms of query response time.

5. RESULT AND ANALYSIS

An experiment was carried out on IBM 3400X Server running on Xeon Processor at 2.3 GHz and having 12GB of RAM. The BenchmarkFactory, a load generation and analysis tool was used for generating user load and measuring response times.

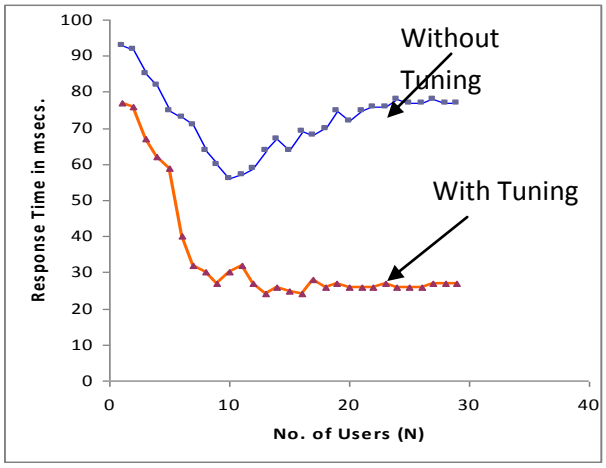

Figure 6. Response Time V/s No. of Users

A standard TPC-C type workload representing an OLTP application scenario was chosen. The experiment was carried out once without the fuzzy-controlled system and once with fuzzy controlled system in place, by proactively monitoring the user load and buffer cache size and applying corrections to the estimated buffer cache size and shared pool using the fuzzy inference rules. The fuzzy control system is implemented using MATLAB 7.0.

The Buffer Hit Ratio by definition is given by

BHR = 1- Physical Reads/ (db block gets + consistent gets)

The Buffer Hit Ratio is obtained using the following query

```
SELECT 100*(1-(v3.value/v1.value+v2.value) as "BHR"
FROM    v$sysstat v1, v$sysstat v2, v$sysstat v3
WHERE  v.name='db block gets'
AND      v.name='consistent gets'
AND      v.name='physical reads' ;
```

After the fuzzy controller estimates the required buffer size applying the fuzzy rules to the input fuzzy variables, the tuner module alters the buffer caches size using the following commands. The buffer cache is dynamically altered using the command

ALTER SYSTEM SET db_cache_size=24M scope=both;

and shared pool using

ALTER SYSTEM SET shared_pool_size=44M scope=both;

As shown in Fig. 6, in the initial part of the graph, the response time is very high. This is due to the fact that, initially majority of the data blocks requested by the users queries are not in the buffer cache, resulting in increased level of disk activity. But as more and more users query the database using similar or already executed queries, the response time decrease as is evident from the initial part of the graph. However, for an un-tuned DBMS the response time rises rapidly beyond a user load of 10 and the response time is not smooth.

As can be seen from the results in Fig. 6, that fuzzy tuning has significant impact over the performance of the DBMS. As a result of smooth tuning effect, the response time stays fairly stable even when the number of users increases from 10 to 30. This shows the control action of the fuzzy control block of the proposed architecture is quite successful in keeping the response time quite flat after 16 users.

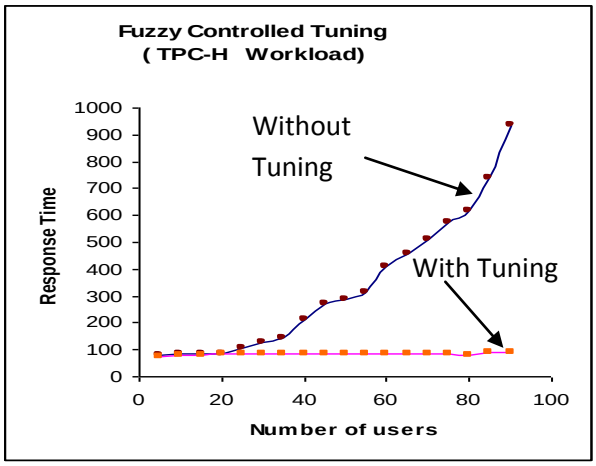

Figure 7. Response Time V/s Users

Fig. 7 shows the response time for TPC-H work load that represents a Decision Support System. As can be seen from the graph, the tuning effect is very effective. The response time is almost flat over the entire range of number of users. This result is very much on the expected lines for TPC-H kind of workload.

6. CONCLUSIONS

The trend in database tuning is towards building self tuning systems employing conventional and adaptive controls, comprising self learning architectures that are effective and provide stable response times. In this paper, a new control architecture based on Fuzzy Control is presented. The system is tested with two workload types, namely, OLTP and DSS workload types and the results show, significant improvement in Query Response times even when the system is subjected to different load types. However, further research is needed to study the behavior of the system when presented with

transient and web workloads and improve the ability of the system to self tune, even more efficiently and effectively.

7. ACKNOWLEDGMENTS

We sincerely acknowledge the contributions of Mr. Benson Fernandes and Mrs. Anita Kenchennavar in carrying out the load testing and generating the reports. We also sincerely thank our colleague Prof. S.A.Kulkarni for scrutinizing the paper. Our thanks are also due to our esteemed Management for their support. Our Sincere thanks to Computer Center Head Prof. S.R.Mangalwede for providing us with the computing facilities.

8. REFERENCES

[1] S. Agarwal and et.al., "Automated Selection of Materialized Views and Indexes", Conference Proceedings, 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.

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

[5] Yi-Cheng Tu, Gang Ding, "Control-Based Tuning under Dynamic Workloads, IEEE Conference 2007.

[6] Rimma V. Nehme, "Database, Heal Thyself", Data Engineering Workshop April 2008.

[7] C.C.Lee, "Fuzzy Logic in Control Systems", IEEE Trans. on Systems, Man, and Cybernetics, SMC, Vol. 20, No. 2, 1990, pp. 404-35

[8] 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 .

[9] Wiese, David; Rabinovitch, Gennadi, "Knowledge Management in Autonomic Database Performance Tuning", 20-25 April 2009.

[10] Ahmed A. Soror, Ashraf Aboulnaga, Kenneth Salem, "Database Virtualization : A new Frontier for Database Tuning and Physical Design", IEEE 2007, pp 388-394.

[11] Stephan Krumpass, Andreas Scholz, Martina, "Quality of Serve Enabled Management of Database Workloads, Bulletin of IEEE Computer Society on Data Engineering, 2008.

[12] Choudhuri and G. Weikum, "Rethinking Database System Architecture: Towards a Self Tuning RISC style Database System", in VLDB, 2000, pp 1-10.

[13] S. W. Cheng, D. Garlan et. al, "Architecture based Self Adaptation in the presence of multiple objectives", Proceedings of 2006 International journal of Computer Systems and Engineering., 2006.

[14] Sanjay Agarwal, Nicolas Bruno, Surajit Chaudhari, "AutoAdmin: Self Tuning Database System Technology", IEEE Data Engineering Bulletin, 2006.

[15] DageVille and K. Dias, "Oracle's self tuning architecture and solutions", IEEE Data Engg. Bulletin, Vol 29, 2006.

[16] Gerhar Weikum, Axel Moenkerngerg et. al., Self-tuning Database Technology and Information Services : From wishful thing to viable Engineering", Parallel and Distributed Information System 1993.

[17] Chaudhuri, S.; Weikum G, "Foundations of Automated Database Tuning", Data Engineering, April 2006.

[18] Satish, S.K.; Saraswatipura, M.K.; Shastry, S.C, "DB2 performance enhancements using Materialized Query Table for LUW Systems", 2007. ICONS '07. Second International Conference, April 2007.

[19] Daniel Manasce, Bruno D., Daniel Barbara, "Fractal Characterization of Web Workloads", 2002.

[20] J. Abanoyi, L. Nyagi, F. Seibefertz, 'Adaptive Sugeno Fuzzy Control : A Case Study ", IEEE Conference 2002.