Vol. 5, No. 10 October 2014 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

Analysis of Server Performance with Different Techniques of Virtual Databases

[1] V.S. Dhaka, [2] Sonali Vyas
[1] Research Guide, School of Computer and Systems Sciences, Jaipur National University
[2] Research Scholar, School of Computer and Systems Sciences, Jaipur National University
[1] vijaypal.dhaka@gmail.com, [2] vyas.sonal86@gmail.com

ABSTRACT
Database virtualization is understood by different people in different ways; either it is executing or storing databases on virtual machine or a full virtualization of database. It helps in masking the actual physical location of the databases from the programs and users using it. With the aid of virtualization, databases can be decentralized thus increasing availability and usability of databases. But as we step into the world of virtualization, the central requirement becomes performance. It becomes an important task for DBA to provide excellent performance by putting minimum efforts and cost. Performance confines various extensions. A database residing in a virtual machine must act commensurable with the application which is using it. It is also considered that resource sharing between multiple virtual machines running on same host must be effective and efficient. In this paper, the performance criteria on the basis of parameters involved in running queries on virtual databases are discussed and a report is prepared analyzing the results of different tuning techniques.

Keywords: *Virtualization, Performance tuning, TPC-H, TPC-W, Query Complexity.*

1. INTRODUCTION

Virtualization is an emerging term in IT world. Virtualization provides a common platform for better availability, scalability, manageability and security. These days everything is getting virtualized like server, hardware, Operating Systems etc. Database Virtualization provides provision for masking the physical location of database from the users and their querying programs. Today various companies are providing different techniques and tools like Xen, VMware, Virtual Box etc, for virtualizing the databases. Virtualization builds an adaptable layer of software between applications and their resources. This layer maps virtual resources absorbed by applications to physical resources. Virtualization unambiguously allocates resources to multiple applications. This enables virtualization an efficient tool for easily deploying and managing databases and other software [1]. But in the world of virtualization, performance is an essential requirement. The elementary paradigm of virtualization performance is distinguishing between workloads carried on virtual environment and same workload carried on in non-virtual environment. If the workload is utilizing maximum amount of CPU resources, it is marked as an overhead of virtualization [2]. For tuning database performance in virtual environment some tuning parameters must be set either by DBA or automatically. The necessary parameters are those which regulate the allocation of physical resources like CPU, Memory, etc. These parameters also collaborate with database tuning parameters like buffer sizes, which mean these pair of parameters should be tuned concurrently. Various techniques and benchmarks like TPC-H, TPC-W etc. are being proposed for tuning performance of virtual databases. TPC-H benchmark is a decision-support benchmark containing business oriented queries. Another benchmark TPC-W is a commercial one containing 34 Select queries and 15 UDI (update, delete and insert) queries [3]. As there are numerous techniques available having different working strategies, therefore it becomes difficult for any database administrator to select among them.

Our aim in this paper is to analyze the outcome of two different techniques of performance tuning and present a result on the basis of their comparison. These techniques are:

a. Optimizing cost using search algorithm for tuning virtual databases [2].
b. Performance prediction model based on algorithmic complexity of queries [3].

2. RELATED WORK

Virtualization can be implemented by various technologies like Xen hypervisor, VMware, Virtual Box, etc. The main aim of virtualization is to carry out multiple workloads on same host simultaneously. Many software and hardware techniques are developed to overcome the problems related to the performance of virtualization so that it can be easily accepted by rapid IT world. Many important dimensions to virtualization are available, which aids in consolidating multiple workloads onto the same server [1].

Various techniques are proposed for tuning the database performance in virtual domain. Virtualization proposes some new tuning parameters which influence the performance of database. The significance of tuning these parameters becomes evident when multiple database system execute in same virtual machine sharing physical resources [2]. Not only the main concern is resource allocation but major overhead in virtualization is cost. Sometimes it becomes the major issue in virtualization performance that, how much cost of manageability is incurred by running database in virtual domain [3]. Another overhead includes time consumption by query execution for multiple users. This is easily evaluated by estimating the time complexities of different queries. It is achieved by a performance-sensitive query locating process on three benchmarks: RUBiS, RUBBoS, and TPC-W [4]. The cost of executing the given transaction is estimated by database query optimization techniques. This gives a concept of using queuing network model to tune performance of database design [5]. One can use many techniques for tuning databases in virtual environment like, by using basic Operating System's utilities [6], proposing self-tuning methods [7], and by proposing a management framework (VEMan) for virtual cluster system and presenting automatic performance tuning technique [9]. Database systems are tuned for utilization of resources which has been priority of many works. [10]. Sometimes the key ingredients of performance are examined and treated as input to the script and buffer cache is estimated by trained networks. Many algorithms like rate change algorithms are used to change the parameters of tuning [11]. For proposing self-tuning methods for a database is a big task as sometimes it slow down the overall progress of system. The only reason behind this problem is the complexity of internal components that needs to be fine-tuned for performance for a powerful language such as SQL [12]. One of the most prominent examples of resource virtualization is machine virtualization, which is the focus of this paper. Machine virtualization technologies, such as Xen [13, 14] or the different VMware products [15], provide the capability of creating different virtual machines that share the resources of one real physical machine (or sometimes even many machines [16] Each one of these virtual machines runs its own separate operating system, and the operating system and applications in each virtual machine perceive that they are using dedicated machine resources (CPU, memory, and I/O), whereas in reality the physical resources are shared among all the virtual machines. The virtualization layer, known as the virtual machine monitor [17, 18], controls the allocation of physical resources to virtual machines and can change it as needed. The virtualization layer also provides other capabilities such as the ability to save an image of a running virtual machine, and to restart the virtual machine from a saved image on the same physical machine or a different one. Some virtualization layers even provide the ability to migrate live, running virtual machines between physical machines [19, 20].

3. PERFORMANCE TUNING TECHNIQUES

3.1 Cost Optimizing Search Algorithm

Besides the benefits of virtualization, a question of resource allocation arises, this is considered as virtualization design overhead. The virtualization design problem can be stated as follows: "Given N database workloads that will run on N database systems inside virtual machines, how should we allocate the available resources to the N virtual machines to get the best overall performance?" Cost incurred in running each workload, Wi, which depends on allocation of resources, Ri, This cost is denoted by Cost (Wi, Ri) and goal is to minimize the overall cost. To attain this goal a search algorithm like greedy method or dynamic programming is implemented and a method is derived for evaluating cost. By using the optimizer of database, cost of workload is estimated in virtual machine. The values of parameters used by optimizer are estimated. The performance is boost up by using index set of TPC-H benchmark. Two workloads TPC-H Query 4 and the other consisting of TPC-H Query 13 are considered. By allocating CPU for these workloads it is seen that performance of Query 4 is not changing significantly as compared to that of Query 13. When multiple copies of workloads are included for equal share of CPU then close values for execution time is obtained. But when 75% CPU allocation is done for Query 13 workloads then performance is increased by 30% without affecting performance of Query 4 as shown in Fig-1.

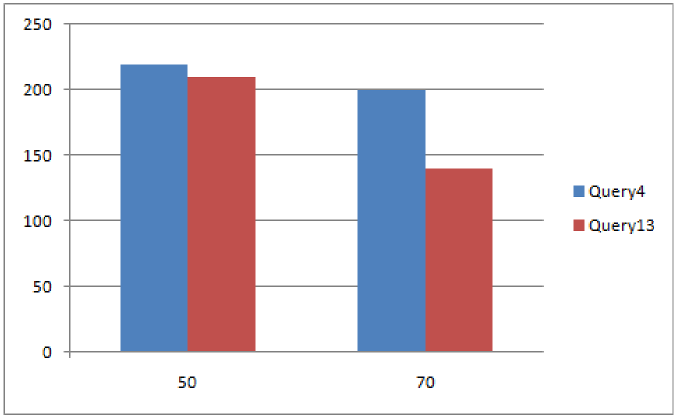

Fig 1: CPU allocation for Query 4 and Query 13

3.2 Performance Prediction Model Based On Algorithmic Complexity Of Queries

All queries are significant for evaluation of performance. Performance prediction model is derived for showing the association between query execution time and database size. This relationship is described by the time complexity of query like O (1), O (log n), O (n log n) and O (n2). Queries with complexity O (1) or O (log n) have less effect on performance. When size of database varies at rapid rate, queries with complexity O (n log n) and O (n) needs to be optimized for better performance. The performance sensitive query locating activity is then explained on three benchmarks: RUBiS, RUBBoS and TPC-W. Among these three, TPC-W is more challenging therefore experiments were executed using it. The performance-sensitive queries are estimated using complexity model. The three performance-sensitive queries are considered, queries 6, 7 and 8, having complexities O(n log(n)) and O(log(n)). All three tables consists of 3900000 tuples approx. and without any loss of generality, 120000 tuples are stated as basic Scale (scale 1). Then, for scales 5,10,15,20,25 and 30 average query run time is determined, as shown in Fig-2. The parameterized complexity function is used to find out the execution time for scale 30. The NLLS (Non-linear least squares) algorithm is used to estimate the query execution time and error rate of three queries 6-8. If 5 data points are considered for these different scales then error rate for given three queries are: 3.9%, 4.2%, and 9.0% , and if 29 data points are taken into account then error rate is 3.48%, 2.74%, and 12.5%. Since, performance sensitive queries have time complexities of O (n log(n)) and O(log(n)), by using the function log(n) and its property that as n increases, the function becomes constant. Thus the complexity function y=a log(n)+b nlog(n)+c can be presented as y=a n+b. Accordingly, the data points for scales 10, 15, 20 and 25 are used to suit the complexity function y=a n+b for determining the query run time at scale 30. Applying this strategy, the error rates of queries 6-8 were 2.3%, 2.4% and 4.2%.

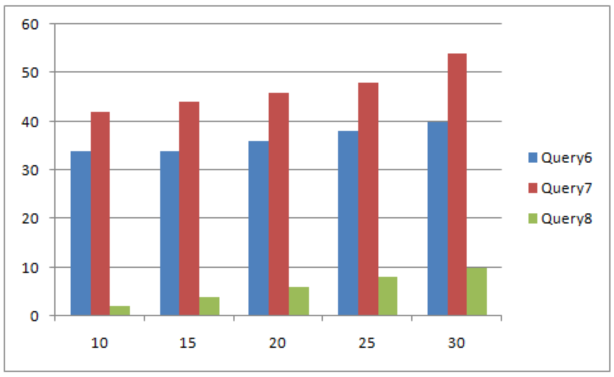

Fig 2: Query Runtime for Query 6, Query7 and Query8

4. ANALYSIS AND COMPARISON

As discussed above, both techniques are for performance tuning of database running in multiple query environment. But as we move into virtual environment then performance becomes an important issue to manage. In this paper, it is seen that the Technique-I can be effectively used to normalize the virtualization problem with the help of cost model in identifying resource allocation among workloads. It is also analyzed that there can be efficient allocation of resources than by simply allocating equal share of resources to the workloads. Many such techniques involve black-box models to determine resource utilization. However, it is assumed by many models that size of database does not vary with change in workloads. But in Technique-II, database sizes variations are studied for analyzing performance variations. In this technique, query complexity functions are used to identify the association between query run-time and database size. This shows that all performance sensitive queries have complexities of O (n log (n)) and O (n). Therefore, the analysis of performance-sensitive queries is carried by using the property of function Log (n), according to their complexities. At the scale of 5, 10, 15, 20 and 25 the average execution time is predicted. The predicted error rates for the three queries 6-8 are 2.3%, 2.4%, and 4.2% with this method. By this analysis, the two techniques are reviewed and its result is as mentioned in Table 1.

Table 1: Comparison of technique-I and technique-II

Analyzed Fields	Technique I	Technique II
Technology used	Benchmark: TPC-H	Benchmark: TPC-W
Base of analysis	Resource Allocation	Complexity of Queries
Workload used	2 workloads: Q4 and Q13	3 workloads: Q6-Q8
Execution time	At 75% CPU allocation: Q4: 210(s) and Q13:150(s)	At scale 30 and for 29 data points, Q6: 42ms, Q7:54ms and Q8: 11ms

5. CONCLUSION

As virtualization approach is followed for database, the major issue is to manage the performance of queries in multi-user environment. In this paper, the analysis of two tuning techniques is explicated and reviewed. For Technique-I, optimization is the main ingredient for cost-modeling approach. This technique elaborates the concept of resource allocation by assigning the maximum resource to complex ones. It defines the relationship between allocation of resources and performance of queries for solving virtualization design problems. While the technique-II concentrates on the database size for improving the performance issues. It also explicates association between query execution time and size of database by analyzing the complexity of queries. But as virtual environment is considered, there is a limitation with Technique-II because in it NLLS algorithm is used to determine average execution time. For this algorithm the relative scale should be realized for the complexity function. This is difficult in virtual environment as service consumers only request for database structure and data is kept as valuable possession and is not dispensed with third party by service providers. Therefore it will be difficult for service provider to calculate the relative scale from database log alone. As concluded this technique is less suitable for virtual environment as compared to Technique-I.

REFERENCES

[1] Soror Ahmed A. Soror Ashraf Aboulnaga Kenneth Salem, “Database Virtualization: A New Frontier for Database Tuning”.

[2] Richard McDougall, Jennifer Anderson, Virtualization Performance”, ACM SIGOPS Operating Systems Review archive Volume 44 Issue 4, December 2010.

[3] Chihung Chi, Ye Zhou, and Xiaojun Ye, “Performance Prediction for Performance-Sensitive Queries Based on Algorithmic Complexity”, TSINGHUA SCIENCE AND TECHNOLOGY, ISSNl l1007 0214ll08/10llpp618-628, Volume 18, Number 6, December 2013.

[4] Umar Farooq Minhas, Jitendra Yadav, Ashraf Aboulnaga and Kenneth Salem, “Database systems on virtual machines: How much do you lose?” In Data Engineering Workshop, 2008.

[5] Rasha Osman, Irfan Awan, Michael E.Woodward, “Application of Queuing Network Models in the Performance Evaluation of Database Designs”, Electronic Notes in Theoretical Computer Science (ENTCS) archive, Volume 232, March, 2009.

[6] Hitesh KUMAR SHARMA, Aditya SHASTRI, Ranjit BISWAS, “A Framework for Automated Database Tuning Using Dynamic SGA Parameters and Basic Operating System Utilities”, Database Systems Journal vol. III, no. 4/2012.

[7] Surajit Chaudhuri, Vivek Narasayya, “Self-Tuning Database Systems: A Decade of Progress”, Microsoft Research, 2007.

[8] Biplob K. Debnath , David J. Lilja , Mohamed F. Mokbel, “SARD: A Statistical Approach for Ranking Database Tuning Parameters”, Data Engineering Workshop, 2008.

[9] Rasha Osman, William J. Knottenbelt, “Database system performance evaluation models: A survey”, Performance Evaluation archive Volume 69 Issue 10, October, 2012

[10] G. Weikum, A. M ̈onkeberg, C. Hasse, and P. Zabback. Self- tuning database technology and information services: from wishful thinking to viable engineering. In Proc. Int. Conf. on Very Large Data Bases (VLDB), 2002

[11] Ankit Verma, “Enhanced Performance of Database by Automated Self-Tuned Systems”, JCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 01, May 2011.

[12] Surajit Chaudhuri, Vivek Narasayya, “Self-Tuning Database Systems: A Decade of Progress”, Microsoft Research, 2007.

[13] P. T. Barham, B. Dragovic, K. Fraser, S. Hand, T. L. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In Proc. ACM Symposium on Operating Systems Principles (SOSP), 2003.

[14] XenSource. http://www.xensource.com/.

[15] VMware. http://www.vmware.com/.

[16] Virtual Iron. http://www.virtualiron.com/

[17] M. Rosenblum and T. Garfinkel. Virtual machine moni-tors: Current technology and future trends. IEEE Computer, 38(5), 2005

[18] J. E. Smith and R. Nair. The architecture of virtual machines. IEEE Computer, 38(5), 2005

[19] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In Proc. Symposium on Networked Systems Design and Implementation (NSDI), 2005

[20] C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum. Optimizing the migration of virtual computers. In Proc. Symposium on Operating System De-sign and Implementation (OSDI), 2002