International Journal of Advanced Scientific Research
ISSN: 2456-0421; Impact Factor: RJIF 5.32
www.allscientificjournal.com
Volume 2; Issue 2; March 2017; Page No. 66-70

Deploying any type of database appliances in the cloud world

Dr. I Lakshmi

Assistant Professor, Department of Computer science, Stella Maris College, Chennai, Tamil Nadu, India

Abstract
Distributed computing is an undeniably prevalent worldview for getting to processing assets. A well known class of registering mists is Infrastructure as a Service (IaaS) mists, exemplified by Amazon's Elastic Computing Cloud (EC2). In these mists, clients are offered access to virtual machines on which they can introduce and run self-assertive programming, including database frameworks. Clients can likewise convey database apparatuses on these mists, which are virtual machines with pre-introduced pre-arranged database frameworks. Sending database machines on IaaS mists and execution tuning and streamlining in this environment present some fascinating exploration challenges. In this paper, we introduce some of these difficulties and we layout the devices and strategies required to address them. We introduce a conclusion to-end answer for one tuning issue in this environment, specifically dividing the CPU limit of a physical machine among various database apparatuses running on this machine. We additionally diagram conceivable future research headings around there.

Keywords: cloud world, database appliances, Amazon's Elastic Computing Cloud

Introduction

Cloud computing has emerged as a powerful and cost-effective paradigm for provisioning computing power to users. In the cloud computing paradigm, users use an intranet or the Internet to access a shared computing cloud that consists of a large number (thousands or tens of thousands) of interconnected machines organized as one or more clusters. This provides significant benefits both to providers of computing power and to users of this computing power. For providers of computing power, the push to cloud computing is driven by economies of scale. By operating massive clusters in specially designed and carefully located data centers, providers can reduce administrative and operating costs, such as the costs of power and cooling [15, 16]. In addition, the per-unit costs of hardware, software and networking become significantly cheaper at this scale [4]. For users, cloud computing offers simple and flexible resource provisioning without up-front equipment and set up costs and on-going administrative and maintenance burdens. Users can run software in the cloud, and they can grow and shrink the computing power available to this software in response to growing and shrinking load [4]. There are different flavors of cloud computing, depending on how much flexibility the user has to customize the software running in the cloud. In this paper, we focus on computing clouds where the user sees a barebones machine with just an operating system and gets full flexibility in installing and configuring software on this machine. These clouds are known as Infrastructure as a Service (IaaS) clouds. A very prominent example of this type of cloud is Amazon's Elastic Computing Cloud (EC2) [2], which enables users to rent computing power from Amazon to run their software. Other providers of this style of cloud computing include GoGrid [13] and App Nexus [3]. Additionally, many organizations are building IaaS clouds for their internal use [6, 22]. In IaaS clouds, users are typically given access to virtual machines (VMs) [5, 23] on which they can install and run software. These virtual machines are created and managed by a virtual machine monitor (VMM) which is a layer of software between the operating system and the physical machine. The VMM controls the resources of the physical machine, and can create multiple VMs that share these physical machine resources. The VMs have independent operating systems running independent applications, and are isolated from each other by the VMM. The VMM controls the allocation of physical machine resources to the different VMs. The VMM also provides functionality such as saving and restoring the image of a running VM, or migrating VMs between physical machines.

A common model for deploying software in virtual machine environments is the virtual appliance model. A virtual appliance is a VM image with a pre-installed pre-configured application. Deploying the application simply requires copying this VM image to a physical machine, starting the VM, and performing any required configuration tasks. The cost of installing and configuring the application on the VM is incurred once, when the appliance is created, and does not need to be incurred again by users of the appliance. A database appliance is a virtual appliance where the installed application is a database system. With the increasing popularity of virtualization and cloud computing, we can expect that a common way of providing database services in the future will be through database appliances deployed in IaaS clouds. As an example of this deployment mode, Amazon offers MySQL, Oracle, and Microsoft SQL Server virtual appliances for deployment in its EC2 cloud. An important question to ask is how to get the best database system performance in this environment. Cloud providers are interested in two related performance objectives: maximizing the utilization of cloud resources and minimizing the resources required to satisfy user demand. Users are interested in minimizing application response time or maximizing application throughput. Deploying database appliances in the cloud and tuning the database and virtualization parameters to optimize performance introduces some interesting research challenges. In this paper, we outline some of these challenges (Section 2), and we

present the different tools and techniques required to address them (Section 3). We present our work on partitioning CPU capacity among database appliances as an example end-to-end tuning solution for virtualized environments (Section 4). We conclude by outlining some possible future research directions in this area (Section 5).

2 Deployment and Tuning Challenges

Our focus is on deploying and tuning virtual machines running database systems (i.e., database appliances) on large clusters of physical machines (i.e., computing clouds). This raises deployment and computing challenges, which we describe next.

2.1 Deployment Challenges

Creating a database appliance that can easily be deployed in a cloud, and obtaining an accessible, usable database instance from this appliance require addressing many issues related to deployment. These issues are not the research focus of our work, but we present them here since these seemingly simple and mundane tasks can be very tricky and time consuming. These issues include:

Localization

When we start a VMfrom a copy of a database appliance, we need to give this new VMand the database system running on it a distinct "identity." We refer to this process as *localization*. For example, we need to give the VM a MAC address, an IP address, and a host name. We also need to adapt (or localize) the database instance running on this VM to the VM's new identity. For example, some database systems require every database instance to have a unique name, which is sometimes based on the host name or IP address. The VMM and the underlying operating system and networking infrastructure may help with issues such as assigning IP addresses, but there is typically little support for localizing the database instance. The specific localization required varies from database system to database system, which increases the effort required for creating database appliances.

Steering

Notwithstanding giving each VM and database occasion an unmistakable personality, we should have the capacity to course application solicitations to the VM and database case. This incorporates the IP-level directing of parcels to the VM, however it additionally incorporates ensuring that database solicitations are steered to the right port and not hindered by any firewall, that the show is directed back to the customer reassure if necessary, that I/O solicitations are steered to the right virtual stockpiling gadget if the "process" machines of the IaaS cloud are not quite the same as the capacity machines, etc.

Verification:

The VM must know about the accreditations of all customers that need to associate with it, free of where it is keep running in the cloud.

2.2 Tuning Challenges

Next, we turn our regard for the difficulties identified with tuning the parameters of the virtualization environment and the database machine to accomplish the craved execution destinations. These are the essential concentration of our examination work, and they include:

Arrangement

Virtualization permits the cloud supplier to run a client's VM on any accessible physical machine. The mapping of virtual machines to physical machines can significantly affect execution. One basic issue is to choose what number of virtual machines to keep running on each physical machine. The cloud supplier might want to limit the quantity of physical machines utilized; however running more VMs on a physical machine corrupts the execution of these VMs. It is critical to adjust these clashing goals: limiting the quantity of physical machines utilized while keeping up satisfactory execution for clients. A more refined mapping of virtual machines to physical machines could consider not just the quantity of VMs per physical machine, additionally the asset necessities of these VMs. The situation calculation could, for instance, abstain from mapping numerous I/O serious VMs to the same physical machine to limit I/O impedance between these VMs. This kind of mapping requires understanding the asset utilization attributes of the application running in the VM, which might be less demanding to accomplish for database frameworks than for different sorts of uses since database frameworks have a very adapted and regularly unsurprising asset use design.

Asset Partitioning

Another tuning test is to choose how to segment the assets of each physical machine among the virtual machines that are running on it. Most VMMs give devices or APIs to controlling the way that physical assets are assigned. For instance VMM booking parameters can be utilized to distribute the aggregate physical CPU limit among the VMs, or to control how virtual CPUs are mapped to physical CPUs. Other VMM parameters can be utilized to control the measure of physical memory that is accessible to each VM. To get the best execution, it is valuable to consider the attributes of the application running in the VM with the goal that we can distribute assets where they will give the most extreme advantage. Database frameworks can profit by this application-educated asset apportioning, as we will appear in Section 4.

Benefit Level Objectives

To enhance the execution of a database machine in a cloud domain, it is useful to have the capacity to express extraordinary administration level targets. The abnormal state tuning objective is to limit the cloud assets required while keeping up satisfactory execution for the database apparatus. Communicating this idea of "satisfactory execution" is not a minor errand. A database framework is normally part of a multi-layer programming stack that is utilized to serve application demands. Benefit level understandings are commonly communicated as far as end-to-end application execution, with no sign of the amount of this execution spending plan is accessible to the database framework versus what amount is accessible to different layers of the product stack (e.g., the application server and the web server). Determining the execution spending that is accessible to the database framework for a given application demand is difficult, since an application demand can bring about a shifting number of database solicitations, and these database solicitations can differ extraordinarily in multifaceted nature relying upon the

SQL articulations being executed. Tuning in a cloud domain in this way requires creating viable and instinctive methods for communicating database benefit level destinations. Diverse workloads can have distinctive administration level targets, and the tuning calculations need to consider these distinctive administration level goals.

Powerfully Varying Workloads

Tuning the execution of a database machine (e.g., position and asset apportioning) requires information of the apparatus' workload. The workload can essentially be the full arrangement of SQL proclamations that execute at the apparatus. Nonetheless, it is a fascinating inquiry whether there can be a more compact yet helpful representation of the workload. Another intriguing inquiry is whether some tuning choices can be made without information of the SQL explanations (e.g., if this is another database case). It is additionally critical to recognize when the way of the workload has changed, potentially by arranging the workload [11] and distinguishing when the workload class has changed. The tuning calculations should have the capacity to manage powerfully changing workloads that have distinctive benefit level targets.

3 Tools and Techniques

Next, we turn our consideration regarding the apparatuses and procedures that are expected to address the tuning challenges sketched out above. These include:

Execution Models

Foreseeing the impact of various tuning activities on the execution of a database machine is a basic segment of any tuning arrangement. This requires creating precise and effective execution models for database frameworks in virtualized situations. There are two general classes of models: white box models, which depend on interior learning of the database framework, and discovery models, which are ordinarily factual models in view of outer, exact perceptions of the database framework's execution. White box demonstrating is particularly appealing for database frameworks for two reasons. In the first place, database frameworks have an adapted and obliged interface for client demands: they acknowledge and execute SQL proclamations. This streamlines characterizing the contributions to the execution demonstrate. Second, and all the more critically, database frameworks as of now have very refined inner models of execution. One approach to construct a white box model is to uncover these inside models to the tuning calculation and adjust them to the tuning job that needs to be done. For instance, the question enhancer cost demonstrate, which has been utilized broadly as a consider the possibility that cost show for programmed physical database plan [7], can be utilized to evaluate the impact of apportioning diverse shares of physical assets to a database. machine (see the following segment for more subtle elements). Self-overseeing database frameworks have other inner models that can be uncovered for use in execution tuning in a cloud domain. These incorporate the memory utilization display utilized by a self-tuning memory director [8, 21] or the model utilized for programmed analysis of execution issues [10].

The disservice of white box displaying is that the required execution models don't generally exist in the database framework, and creating white box models without any preparation is troublesome and tedious. Notwithstanding when inward models do exist in the database framework, these models are in some cases not aligned to precisely give the required execution metric, and they once in a while make streamlining presumptions that disregard critical parts of the issue. For instance, the question enhancer cost model is composed fundamentally to look at inquiry execution arranges, not to precisely evaluate asset utilization. This cost show concentrates on one question at any given moment, disregarding the occasionally noteworthy impact of simultaneously running connecting inquiries [1]. On account of these weaknesses of white box displaying, it is some of the time alluring to manufacture discovery models of execution by fitting factual models to the watched after effects of execution tests [1]. When fabricating these models it is vital to painstakingly choose which execution tests to lead to gather tests for the model, since these investigations can be exorbitant and they considerably affect demonstrate exactness [18]. Be that as it may, the figment of unbounded processing assets gave by IaaS mists can really rearrange black box exploratory displaying of database frameworks, since we can now effectively arrangement the same number of machines as we have to run the execution tests required for building a precise model. An intriguing exploration question is whether it is conceivable to join the best components of black box and white box demonstrating, by utilizing the inward models of the database framework as a beginning stage, yet then refining these models in view of exploratory perceptions [12].

Enhancement and Control Algorithms:

Taking care of the execution tuning issues of a cloud domain requires creating combinatorial advancement or programmed control calculations that utilization the execution models depicted above to settle on the best tuning activity. These calculations can be static calculations that accept a settled workload, or they can be powerful calculations that adjust to evolving workloads. The calculations can just have as an objective the best-exertion augmentation of execution [20], or they can plan to fulfil distinctive administration level destinations for various workloads [17].

Instruments for System Administrators:

Notwithstanding the models and calculations depicted above, framework managers require instruments for sending and tuning database machines. These instruments ought to uncover the execution qualities of the VM, as well as the execution attributes of the database framework running on this VM. For instance, it is helpful to uncover the imagine a scenario in which execution models of the database framework to framework chairmen so they can make educated tuning.

5 Future Directions

The previous section illustrates a simple performance tuning problem in a cloud computing environment and its solution. Extending the research outlined in the previous section opens up many possibilities for future work, which we are exploring in our ongoing research activities. Instead of partitioning the resources of one physical machine among the VMs, we can consider multiple physical machines and partition their resources among the VMs, that is, decide which physical machine to use for each VM and what share of this machine’s resources are

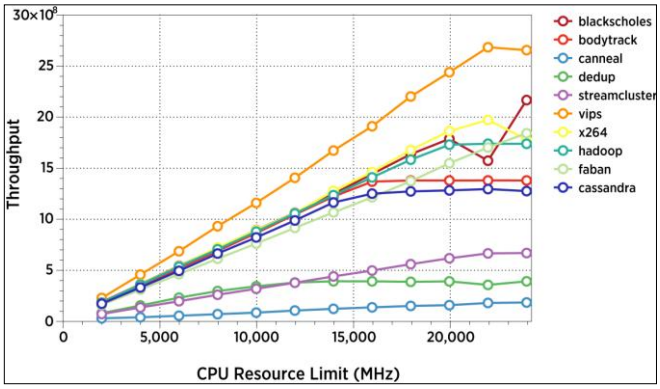

Fig 1: Effect of varying CPU allocation on workload performance.

Allocated to the VM. We can also extend the work to deal with dynamically varying workloads, possibly with different explicit service level objectives. Another interesting research direction is improving the way we refine the query-optimizer-based cost model in response to observed performance.

Another interesting research direction is optimizing the allocation of I/O resources to different VMs. Some VMMs, such as VMWare ESX server [23], provide mechanisms for controlling how much of the I/O bandwidth of a physical machine is allocated to each VM running on this machine. Another mechanism to control the allocation of I/O resources to VMs is controlling the mapping of VM disks to physical disks. Using these two mechanisms to optimize the performance of database appliances is an interesting research direction, especially since many database workloads are I/O bound.

It would also be interesting to explore whether we can expose internal database system models other than the query optimizer cost model and use these models for tuning VM parameters or co-tuning VM and database system parameters. For example, the memory manager performance model can be used to control memory allocation. The cloud environment also offers new opportunities, beyond the challenges of tuning database appliances. For example, since we can provision VMs on-demand, it would be interesting to explore the possibility of scaling out a database system to handle spikes in the workload by starting new replicas of this database system on newly provisioned VMs. This requires ensuring consistent access to the database during and after the replication process, coordinating request routing to the old and new VMs, and developing policies for when to provision and de-provision new replicas. Finally, this idea of application-informed tuning of the virtualized environment is not restricted to database systems. This idea can be used for other types of applications that run in a cloud environment, such as large scale data analysis programs running on Map-Reduce style platforms [7, 9].

6 Conclusion

As cloud computing becomes more popular as a resource provisioning paradigm, we will increasingly see database systems being deployed as virtual appliances on Infrastructure as a Service (IaaS) clouds such as Amazon's EC2. In this paper, we outlined some of the challenges associated with deploying these appliances and tuning their performance, and we discussed the tools and techniques required to address these challenges. We presented an end-to-end solution to one tuning problem, namely partitioning the CPU capacity of a physical machine among the database appliances running on this machine. We also described some future directions for this research area. It is our belief that the style of application-informed tuning described in this paper can provide significant benefits to both providers and users of cloud computing.

References

1. Mumtaz Ahmad, Ashraf Aboulnaga, Shivnath Babu, Kamesh Munagala. Modeling and exploiting query interactions in database systems. In Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM). 2008.
2. Amazon EC2. http://aws.amazon.com/ec2/.
3. App Nexus. http://www.appnexus.com/.
4. Michael Armbrust, Armando Fox, Rean Griffith, Anthony Joseph D, Randy Katz H, Andrew Konwinski *et al.* Above the clouds: A Berkeley view of cloud computing. Technical report, EECS Department, University of California, Berkeley. 2009.
5. Paul Barham T, Boris Dragovic, Keir Fraser, Steven Hand, Timothy Harris L, Alex Ho, Rolf Neugebauer *et al.* Xen and the art of virtualization. In Proc. ACM Symp. on Operating Systems Principles (SOSP). 2003.
6. Luiz Andr´e Barroso, Jeffrey Dean, Urs H¨olzle. Web search for a planet: The Google cluster architecture. IEEE Micro. 2003.
7. Surajit Chaudhuri, Vivek R. Narasayya. An efficient cost-driven index selection tool for Microsoft SQL Server. In Proc. Int. Conf. on Very Large Data Bases (VLDB). 1997.
8. Benoˆıt Dageville, Mohamed Za¨ıt. SQL memory management in Oracle9i. In Proc. Int. Conf. on Very Large Data Bases (VLDB). 2002.
9. Jeffrey Dean, Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. In Proc. Symp. On Operating System Design and Implementation (OSDI). 2004.
10. Karl Dias, Mark Ramacher, Uri Shaft, Venkateswara Venkataramani, Graham Wood. Automatic performance diagnosis and tuning in Oracle. In Proc. Conf. on Innovative Data Systems Research (CIDR). 2005.
11. Said Elnaffar, Patrick Martin, Randy Horman. Automatically classifying database workloads. In Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM). 2002.
12. Archana Ganapathi, Harumi Kuno, Umeshwar Dayal, Janet Wiener, Armando Fox, Michael Jordan *et al.* predicting multiple metrics for queries: Better decisions enabled by machine learning. In Proc. IEEE Int. Conf. on Data Engineering (ICDE). 2009.
13. GoGrid. http://www.gogrid.com/.
14. Hadoop. http://hadoop.apache.org/.
15. James R. Hamilton. Cost of power in large-scale data centers, Nov 2008.http://perspectives.mvdirona.com/2008/11/28/CostOfPowerInLargeScaleDataCenters.aspx.
16. Randy H. Katz. Tech titans building boom. IEEE Spectrum. 2009.
17. Pradeep Padala, Kang Shin G, Xiaoyun Zhu, Mustafa Uysal, Zhikui Wang, Sharad Singhal *et al.* Adaptive control of virtualized resources in utility computing environments. In Proc. European Conf. on Computer Systems (EuroSys). 2007.

18. Piyush Shivam, Varun Marupadi, Jeffrey Chase S, Thileepan Subramanian, Shivnath Babu. Cutting corners: Workbench automation for server benchmarking. In Proc. USENIX Annual Technical Conference. 2008.
19. Ahmed Soror A, Ashraf Aboulnaga, Kenneth Salem. Database virtualization: A new frontier for database tuning and physical design. In Proc. Workshop on Self-Managing Database Systems (SMDB). 2007.
20. Ahmed Soror A, Umar Farooq Minhas, Ashraf Aboulnaga, Kenneth Salem, Peter Kokosielis, Sunil Kamath. Automatic virtual machine configuration for database workloads. In Proc. ACM SIGMOD Int. Conf. on Management of Data. 2008.
21. Adam Storm J, Christian Garcia-Arellano, Sam Light stone, Yixin Diao, Maheswaran Surendra. Adaptive self-tuning memory in DB2. In Proc. Int. Conf. on Very Large Data Bases (VLDB). 2006.
22. Virtual Computing Lab. http://vcl.ncsu.edu/.
23. VMware. http://www.vmware.com/.