Resource Management Strategy for Replication and Migration

Khin Swe Swe Myint
University of Computer Studies, Yangon
khinsweswemyint@gmail.com

Abstract

Data centre virtualization creates an agile environment for application deployment. Applications run within one or more virtual machines and are hosted on various servers throughout the data centre. It improves resource utilization through server consolidation, but it also makes resource management more complex. However, due to the uncertainty of workload variation, the workloads may be unbalanced in the virtualized system, leads to application's SLA violation. In order to deal with such problem, the mathematical expectation approach is used to determine the hotspots in virtualized server. The system expresses the VM migration and replication to relocate virtual machines (VMs) from overloaded servers to underloaded ones.

1. Introduction

Virtualization is an effective approach to enhance resource utilization on demand in cluster server. How to manage the resource effectively to satisfy the Service Level Agreement (SLA) of applications in the virtualized cluster is a complex task. There are various virtualization system can be used cluster server, such as XEN, KVM, VMware, and Virtual PC etc [1]. The virtualization system can support consolidation of multiple application services on a same server and enables performance isolation among different applications.

With increasing scale and complexity of modern enterprise data centers, administrators are being forced to rethink the design of their data centers. In a traditional data center, application computation and application data are tied to specific servers and storage subsystems that are often over-provisioned to deal with workload surges and unexpected failures. Migration can shorten the job completion time by reassigning jobs to the underutilized machines [7]. However, the residual dependency problem with process migration hindered migrating jobs in practice. By decoupling an operating system instance from underlying hardware, server virtualization allows migration with negligible down-time of a virtualized server, also known as live migration [2].

Requests for the virtual server are balanced between the two instances. This should reduce the computing resources needed by a single physical machine by distributing requests to two different virtual machines on two different physical machines. Replication in this work is not an actual copy of the virtual server running at the time, but an instantiation of an image of the virtual server.

Today, there is significant interest in developing more agile data centers, in which applications are loosely coupled to the underlying infrastructure and can easily share resources among themselves. Since applications need to operate above a certain performance level specified in terms of a service level agreement (SLA), effective management of data center resources while meeting SLAs is a complex task. The maximum capacity is seldom reached and results in unused space and wasted resources.

An important characteristic for a well managed data center is its ability to avoid hotspots. Overloaded node often leads to performance degradation and is vulnerable to failures. To alleviate such hotspots, load must be migrated or replicated from the overloaded resource to an underutilized one. Migration is further complicated by the need to consider multiple resources—CPU, network, and memory—for each application and physical server.

The rest of this paper is organized as follows. It presents some extended motivation and address related work in Section 2. In Section 3, background and system overview is described in detail. It will be followed by evaluation in section 4 and conclusion in section 5.

2. Related Work

Currently, there are many researches about dynamic resource management in virtualization systems. Nathuji [6] presents power efficient mechanisms to control various power management policies. In C. Clark [2] dynamic network-bandwidth adaptation allows migration to proceed with minimal impact on running services, while reducing total downtime to below discernable thresholds. It introduces and analyzes the concept of writable working set, and presents the design, implementation and evaluation of high performance OS migration built on top of the Xen VMM. Menasce et al [4] consider CPU as single metric; it controls the server by dynamic CPU priority allocation, and assigns CPU shares to the various virtual machines by beam-search algorithm.

The Wood [8] proposes a Black-Gray box migration called as sandpiper. It migrates overload VMs to under-utilized nodes when hot spot was detected. A. Gambi, M. Pezze, M. Youong [3] show that SLA protection in a virtualized data center depends on structure and behavior at many abstraction levels, and argue that information required for defining autonomic control strategies can be captured by a set of interrelated models.

Zhu et al. developed a three-controller automated resource management system [9]. The purpose of the system is to enable clients and system administrators to focus on policy settings. The system's design combines three controllers: The node controller reallocates resources among the workloads hosted in a physical node; (ii) The pod controller receives information from each node controller in the pod and triggers migrations with the objective of achieving Quality of Service goals (e.g., avoid resource stress situations) while maximizing resource utilization in the pod; (iii) The pod set controller studies the overall performance of various pods and migrates workloads between pods to improve performance.

A mechanism proposes a resource management system for operating system level virtualized environments. It can be implemented that uses replication as an alternative to migration and compares both mechanisms. This should lead to additional strategies for effectively managing resources.

3. Background and System Overview

In the virtualized server, the load of the Web server is changed dynamically over time. According to the dynamical characteristics of the workload, the virtualization plays its important role through adjusting the resource allocation by dynamical migration. The system should be auto-control and self-learning to allocate resource to virtual machine.

A workload increase can be handled by increasing the resources allocated to a virtual server, if idle resources are available on the physical server, or by replicating or migrating the virtual server to a less loaded physical server. Two phases are present, the first phase is to mark out of the hotspot node, collect and handling with the system information and the quantity of the overloaded resource nodes. The second phase is to determine which virtual machine to replicate or migrate.

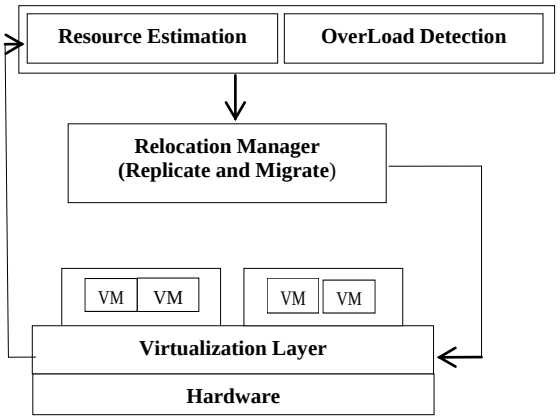

Figure 1. Virtualized System Architecture

In Xen, it implements such architecture. Each virtual server is assumed to be allocated a certain slice of the physical server resources. In the case of CPU, this is achieved by assigning a weight to the virtual server and the underlying Xen CPU scheduler allocates CPU bandwidth in proportion to the weight. In case of the network interface, Xen is yet to implement a similar fair-share scheduler; a best-effort FIFO scheduler is currently used and Sandpiper is designed to work with this constraint. In case of memory, a slice is assigned by allocating a certain amount of RAM to each resident VM.

Figure 1 shows the overview of the virtualized system architecture. It is composed of three components, an estimate resource, load detection and relocation manager. In the estimate resources, it calculates the resources (cpu, net and mem) using the G/G/1 queuing approach and load detection determines the threshold for physical resources. Then relocation manager move the virtual machine to the lower resource utilization physical machine to eliminate the overload. The hardware confi guration of each server—its CPU, network interface and memory characteristics—is assumed to be known to the system. Each physical server runs a virtual machine monitor and one or more virtual machines. Each virtual server runs an application or an application component (the terms virtual servers and virtual machine are used interchangeably). System virtualization provides to the upper layer the abstraction of the underlying hardware — a complete system platform which an operating system can run on. The software layer providing resource virtualization is called virtual machine monitor (VMM). The VMM runs on top of an operating system, while virtual machines (VMs) run on the VMM. Guest operating systems in virtual machines use virtualized resource, while VMM is responsible for mapping virtualized resource to physical resource. Usually, there is privileged domain named domain 0 on VMM which is responsible for managing other VMs and their virtual devices. In a cluster or data center, each physical node runs VMM and one or more VMs.

3.1. Estimate Resources

To estimate peak needs, the peak request arrival rate is first estimated. Since the numbers of serviced request as well as the numbers of dropped request are typically logged, the incoming request rate is the summation of these two quantities. Let λ_{peak} denote the estimated peak arrival rate for the application. An application model is necessary to estimate the peak CPU needs. By using the G/G/1 queuing theory, the system can be captured the results where *d* is the mean response time of requests, *s* is the mean service time. λ_{cap} and λ_{mem} is the request arrival rate. σ^2_a and σ^2_b are the variance of inter-arrival time and the variance of service time, respectively.

$$\lambda \geq \left[s + \frac{\sigma_a^2 + \sigma_b^2}{2.(d-s)} \right]^{-1} \qquad (1)$$

The desired response time *d* is specifi ed by the SLA, the service time *s* of requests as well as the variance of inter-arrival and service times σ^2_a and σ^2_b can be determined from the server logs. λ is the $\lambda_{cap,}$ and λ_{mem}

. They represent the current capacity of the VM. To service the estimated peak workload λ_{peak}, the current CPU capacity needs to be scaled by the factor $\frac{\lambda peak}{\lambda_{cap}}$ and the current memory capacity needs to be scaled by the factor $\frac{\lambda peak}{\lambda_{mem}}$. If the VM is currently assigned a CPU weight w1 and memory weight w2, its allocated share needs to be scaled up by the factor $\frac{\lambda peak}{\lambda_{cap}}$, $\frac{\lambda peak}{\lambda_{mem}}$ to service the peak workload.

The peak network bandwidth usage is simply estimated as the product of the estimated peak arrival rate λ_{peak} and the mean requested file size *b*. The mean request size can be computed from the server logs.

3.2. Load Detection

The Server executes a resource to check on each hardware node. The load detection examines the mathematical expectation of the value of the physical node, for the observation sequence: R_1, $R_2 \ldots R_k$. R represents any resources, and then the mathematical expectation can be expressed：

$$\mu = E(X) = \frac{\sum_{i=1}^{k} R_i}{k} \qquad (2)$$

When the mathematical expectation of observed value exceeds the threshold (75%), the system will be overloaded. In this system, the threshold is a assumption. In the first part, the system has established the threshold of the three kinds of resources and judge whether these physical nodes are hotspots through analyzing the usage of resource.

In the second part, it eliminates hotspots with least physical node.

3.3. Relocation Manager

Dynamic resource management requires monitoring mechanisms and dynamic resource reallocation mechanisms. It was composed of migration and replication.

Migration is an interesting issue for managing resource utilization and performance in clusters. Recent advances in server virtualization have made migration a practical method to achieve these goals.

Replication entails the creation of a replica of a virtual machine on another physical machine. Requests for the virtual server are balanced between the two instances. This should reduce the computing resources needed by a single physical machine by distributing requests to two different virtual machines on two different physical machines.

If there are multiple hotspots, the system can select the physical machine of the largest overloaded resources and move it's virtual machine.

This paper aims at saving performance degradation, and improves the resource utilization while eliminating the hotspot.

4. Evaluation

The web servers were Apache instances and the HTTP requests were generated using httperf. Its were sent to web servers running inside the containers and involved dynamic content so as to increase CPU utilization. The HTTP requests sent to the web servers had an associated timeout of 10 seconds and the time span between the start of two different loads during an experiment was 60 seconds. The metrics used to evaluate the system included lost requests and response time of the web servers. The requests were classified into three categories: lost, failed and successful. A web server's effectiveness was defined as the ratio of the number of successful requests to the total generated requests.

Experiment 1: The managed system consisted of two hardware nodes, HD1 and HD2, and two containers A and B hosted in HD1. A started receiving a load of around 35% (450 requests at a rate of 1 req/sec). After 60 seconds, B started receiving a load of around 52.5% (450 requests at a rate of 1.5 req/sec). At that point in time, HD1 experienced a load of around 87.5%, which exceeded the CPU utilization threshold of 75%. Thus, HD1 was higher resource utilization.

There are three stages; stage1 is the result of the monitoring the resource utilization, stage2 was to search the replications upon detection of the overloaded resources and stage3 was also to search the migrations upon detection of the overloaded resources.

Firstly in stage2, A and B were replicated in HD2 with A' and B' as consequence of two different overloaded resources that were detected in HD1. The web server Srv1.com hosted in A and A', it had 4 failed requests out 450, which resulted in an effectiveness of 99.11%.

The web server Srv2.com hosted in B and B', had 7 failed requests out of 450, which resulted in an effectiveness of 98.44%. B was migrated to HD2 when the overloaded resource was detected in HD1.

Secondly in stage3, the system migrated A to HD2 when overloaded resource situation was detected in HD1.

The web server Srv1.com, hosted in A, had 10 failed requests and 89 lost requests out of 450, which resulted in an effectiveness of 8%. The web server Srv2.com, hosted in B, had 70 lost quests out of 450, which resulted in an effectiveness of 84.44%. Table 1 shows the result of the web server effectiveness in experiment1.

Table 1. The effectiveness of the web server in experiment1

Servers	Stage1	Stage2	Stage3
Srv1.com	100%	99.11%	78%
Srv2.com	62.44%	98.44%	84.44%

Experiment 2: The second experiment was similar to the previous one with the exception that both containers received a load of around 50% (450 requests at a rate of

1.5 req/sec) each. As a consequence, the hardware node HD1 was overloaded and was depicted in stage1.

Firstly in stage2, A and B in HD1 with IDs A' and B' were replicated upon detection of the overloaded HD1.The web server Srv1.com, hosted in A and A', had 21 failed requests and 35 lost requests out of 450, which resulted in an effectiveness of 87.55%. The web server Srv2.com, hosted in B and B', had 25 failed requests and 29 lost requests out of 450, which resulted in an effectiveness of 88%.

In the later stage3, it was similar to the stage2 of the experiment1. Table 2 shows the result of the web server's effectiveness in experiment2.

Table 2. The effectiveness of the web server in experiment2

Servers	Stage1	Stage2	Stage3
Srv1.com	77.55%	87.55%	78%
Srv2.com	62.445	88%	84.44%

5. Conclusion

When a hardware node experiences a overloaded resource, some requests will not be satisfied. The relocation manager represents a convenient solution, since they help to reduce the losses. Replication is preferred over migration when the CPU usage is high. If the CPU usage is relatively low then the migration mechanism is used. However the migration and replication cause no performance degradation, so they could be used as preventive actions in case the load was expected to increase.

References

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand,T.Harris, A. Ho, R. Neuge-bauer, I. Pratt, and A. Warfeld, "Xen and the art of virtualization," *In Proceedings of Symposium on Operating Systems Principles (SOSP),* 2003.

[2] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in NSDI, 2005.

[3] A. Gambi, M. Pezze, M. Youong, "SLA Protection Models for Virtualized Data Centers", *In Proceedings of the Workshop on Software Engineering for Adaptive and Self-Managing Systems (pp.10-19). IEEE*, 2009.

[4] C. Hyser, B. McKee, R. Gardner, Brian J, "Autonomic Virtual Machine Placement in the Data Center", HP Laboratories, HPL-2007-189, 2008.

[5] D.A Menasce, and M.N. Bennani, "Autonomic Virtualized Environments", *In proceedings of International Conference on Autonomic and Autonomous Systems, ICAS '0*, 2006.

[6] R. Nathuji and K. Schwan, "VirtualPower: coordinated power management in virtualized enterprise systems", *In proceedings of In 21st Symposium on Operating Systems Principles (SOSP),* 2007.

[7] P. Sanders, N. Sivadasan , M. Skutella "Online Scheduling with Bounded Migration", Mathematics of Operations Research 34, 2 (2009).

[8] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and Gray-box Strategies for Virtual Machine Migration," *In Proceedings of Symposium on Networked Systems Design and Implementation (NSDI),* 2007.

[9] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian, and L. Cherkasova, "1000 islands: Integrated Capacity and Workload Management for the Next Generation Data Center," *In Proceedings of the 2008 International Conference on Autonomic Computing (ICAC'08),* 2008.