www.ijiarec.com

ISSN:2348-2079

Volume-6 Issue-2

International Journal of Intellectual Advancements and Research in Engineering Computations

Multi Constrained Virtual Machine Allocation for Cloud Services

[1]Lisha M., [2]Bhavya K Bharathan,
[1]PG Scholar, [2]Assistant Professor,
Department of M.Tech. (CSE), Cochin College of Engineering and Technology,
Valiyaparambu, Edayur P.O.,Valanchery, Malappuram District, Kerala, India

Email id: lishamuthuvana@gmail.com

Abstract

The cloud resource provisioning is carried out as services. The resources are provided with user requests. The enterprise applications are build with the support of the service components. The Virtual Machine (VM) allocation for the service components are carried out with resource capacity levels. The process of Virtual Machines to the service components is called as service consolidation or Virtual Machine consolidation process. The dependant service components are services that are used in the same application. The network delay and communication cost is reduced with the support of the intelligent service management models and service dependencies.

The enterprise applications are build with services under many hosts and devices. The resource, services and network constraints are considered in the service consolidation process. The non intrusive dependant service component discovery is achieved with CloudScout model. The service component relationship is analyzed with the time series details collected from the historical data maintained under the logs. The privacy preserved service dependencies are identified with log mining models. The service distance estimation is carried out with the weight of resource metric derived from the iEntropy method. The dependant service clustering process is carried out with the hierarchical and iterative k-means (HiKM) algorithm. The virtual machine consolidation operations are performed with queue network and network latency optimization process.

The optimal service dependency discovery and service consolidation operations are integrated with the CloudScout scheme. The sequential dependencies and complimentary dependencies are combined in the service dependency analysis. The service components are assigned to the virtual machines using the Multi Objective Service Consolidation (MOSC) scheme. The resource level constraints are also analyzed in the service consolidation process.

Index Terms: Cloud services, Virtual Maqchine (VM), Resource allocation in clouds, History log analysis and Service dependency.

1. Introduction

Cloud computing architectures have received an increasing attention in recent years due to their great promises. Cloud providers take advantage of virtualization technologies to gain economical revenues from underutilized IT resources. A core management problem at the infrastructure level is the placement of the VMs on physical hosts. The goal is to optimally exploit available host resources, while avoiding severe performance degradation due to aggregated resource consumption of co-located VMs.

Several recent works addressed VM placement are carried out by considering only local physical node resource constraints, namely CPU and memory sharing, and several algorithms have been proposed with different objectives, such as minimizing the number of powered on hosts. To the best of our knowledge, only few proposals, explicitly considered the effects of network requirements and constraints on VM placement. This aspect, typically referred to as network-aware VM placement, is still widely unexplored and there are several open issues, resulting from the following observations. First, from a networking viewpoint, modern data centers use non-trivial topologies that connect physical hosts through multiple paths for the sake of scalability and reliability. Second, they adopt dynamic multi-path routing schemas to exploit the full available bandwidth; hence, traffic demands routed through network paths can dynamically change at run-time. Finally, Cloud data centers host heterogeneous services that result in a large variety of possible run-time traffic patterns. Moreover, this traffic demand suffers from a high variability due to either unpredicted request spikes or service dependant operations, such as database replication.

Virtualization is being deployed in data centers at a rapid pace to consolidate workloads for improved server utilization, for ease of provisioning, configuration management and more generally, for shows one such configuration for a multi-tier ecommerce web application represented by RUBiS. In this case, client requests arrive at the VM running the web server front end and are then forwarded to one of the VMs running application servers, which in turn may request data from a backend VM hosting a database.

2. Related Work

Among mainstream VMM products, a relative efficient I/O performance and a safe execution environment are provided by the driver domain model, represented by Xen [7], [8], KVM and Microsoft's Hyper-V. A main feature of such a drive domain model is to ensure execution safety in the driver domain model by sacrificing I/O performance. Thus, a fair number of research projects have been dedicated to I/O optimization, CPU scheduling, measuring tools, and fault isolation. However, there still lacks of in-depth understanding of performance interference factors that impact the efficiency and effectiveness of resource multiplexing and scheduling among neighbor VMs.

Xen VMM was first introduced by Barham et al. from the University of Cambridge at 2003. Hypervisor layer and privileged driver domain are implemented to ensure better fault isolation effects [2]. Earlier version of Xen VMM adopts "page-flipping" technique to exchange I/O data between driver domain and guest domain. It has been proved that the cost of mapping and unmapping pages was equivalent to the copy cost for large 1500 byte packets, which means page-flipping is less efficient than copy for small packet sizes [5]. To address the problem of I/O efficiency in the driver domain model, K. Mansley et al. proposed "direct I/O" to alleviate the pressure in driver domain by allowing guest domains to access hardware directly. Though the virtualized system can act as the native system in some cases, direct I/O is not considered in our work, as it lacks of dedicated driver domain to perform fault isolation, which is considered to be the most important function for system virtualization.

Some recent researches contributed to optimize grant mechanism. The purpose was to replace page-flipping by grant copy without giving up driver domain. J. R. Santos et al. discussed grant reuse originally in their work [4]. Grant reuse decreases the frequency of mapping/ unmapping and communicating obviously. Subsequently, an enhanced grant mechanism was designed by K. K. Ram et al. in [6] and implemented in [3]. They adopted previous grant reuse, and added an I/O Translation Table (ITT) between driver domain and hypervisor to facilitate guest domains to perform grant invoke and revoke.

Most existing researches mentioned above focused on optimizing I/O efficiency based on physical host running one VM, while only few studies are related to performance isolation among neighbor VMs. D. Gupta et al. implemented XenMon monitor the detailed system-level values of each VM, such as CPU usage, I/O count and execution count [9]. Based on the monitoring results, ShareGuard and SEDF-DC mechanisms are proposed to improve performance isolation. Y. Koh et al. studied the effects of performance interference between two VMs hosted on the same physical platform by collecting the runtime performance characteristics of different types of concrete applications [1]. Through subsequent analysis of collected characteristics, they predicted the performance of some new applications from its workload characteristic values successfully within an average error of 5%. However, our work reported in this paper, to the best of our knowledge, is the first one that provided a dedicated performance interference study on network I/O workloads running in separate VMs under multiple alternative hardware platforms, ranging from a dual-core CPU with small L2 cache platform to single core CPU with large L2 cache to a multi-core CPU platform.

3. Resource Scheduling with Service Dependency

Today's enterprise applications are designed with suites of independently deployable service components. For instance, an enterprise web application commonly comprises a bunch of application servers, web servers, and database servers. We call the service components that belong to the same application as dependent service components. It is quite beneficial to identify the service dependency, since such knowledge provides a basis for intelligent service management to achieve low network latency of the corresponding applications.

Cloud computing technology aims to deliver everything as a service. With cloud computing, hardware and software are provided on demand. It is a common practice that an isolated virtual machine (VM) with appropriate capacity is provided for each service component. In this paper, we follow the assumption that each service component is assigned to an individual VM, which is widely used by the cloud computing community.

The reason we make this simplifying assumption (one service component per VM) is that the performance difference between one service component per VM and multiple service components per VM is relatively small, i.e., the intra-VM communications cost (between the service components) is close to inter-VM communications cost of 0.398 ms. Consequently, the two cases are sufficiently similar for this study. Hence, we will interchangeably use service component consolidation (a way to manage the mapping between VMs and physical servers) and VM consolidation in the rest of this paper.

With the ever-growing scale and complexity of enterprise applications, it is prevalent that a network application or a parallel application consists of thousands of dependent service components . Due to the capacity constraints of an individual physical host (PH), the dependent service components always span several hosts and network devices, making network efficiency a crucial issue. There are two approaches for achieving high network efficiency: (1) tuning the network structure according to the requirements of hosting VMs and (2) designing algorithms to consolidate the hosting VMs to make maximum use of the network. It is a cumbersome task to tune network structure after the infrastructure of a data center is settled. Therefore, it is a feasible and effective solution to consolidate VMs considering network efficiency, resource constraints, and network structure to promote the performance of network applications. In our work, we select VM consolidation as a representing example to evaluate the impact of service dependency on network efficiency. There are many aspects of network efficiency, here we focus on the data transmission cost and data communication cost.

Three network topologies are commonly adopted in cloud data centers: tree, fat-tree, and VL2. The structure of these three topologies, which all employ divide-andconquer strategies. Switches are divided into three layers (access switch, aggregation switch, and root switch), which relieves the data transmission pressure on a single switch . For example, in the tree topology, data is exchanged through the access switch when the source host and destination host are connected with the same access switch, such as host 1 and host 2. The data is transferred through both the access switch and the connected aggregation switch when the source and destination host are connected with different access switches, such as host 1 and host 3. This is a feasible way to reduce the data transmission cost, since the data transmission distance is reduced through locating dependent service components on hosts that are connected by switches at the lower layer.

We evaluated the average communication time in a large data center across two cities in China with mature network infrastructure: Suzhou City and Hangzhou City. Details of the experimental environment will be introduced. According to the experimental results, we find that the average communication time between a pair of VMs that run on different PHs canvary considerably. The communication time between VMs on the same PH is the shortest (0.398 ms), since the data s directly transferred through the memory bus. The time cost between VMs on different PHs, which are connected by the same aggregation switch, is longer (0.604 ms). The worst case (17.2 ms) is when VMs connected by different aggregation switches are located in different cities. The above observations reveal that the communication cost of a network application can be reduced by consolidating its service components on hosts with ashorter communication time. Service dependency discovery is not trivial, which is capable of reducing the data transmission cost and communication cost by guiding to consolidate dependent services. To discover the service dependency, we adopt a log mining approach due to its advantages of generality and privacy protection (non-intrusive). The key challenges in our work are twofold: service component distance calculation and dependent service component clustering. To solve these challenges, we propose a non-intrusive approach named CloudScout, which discovers the dependency by minin system monitoring data.

In this paper, we describe our experiences in designing, implementing, and evaluating CloudScout. We make the following contributions:

1) We propose a method to calculate the distance between service components. We classify the state of a service component as dormant, stable, or active to calculate the correlations precisely. We develop a method named iEntropy to determine the weight of each resource metric dynamically in the service distance calculation.

2) We present an improved clustering algorithm named HiKM (short for hierarchical and iterative k-means). HiKM uses the hierarchical clustering results for the resource usage metrics and network connection number to determine the initial centroids for k-means. It improves the accuracy of service clustering compared to k-means and hierarchical clustering algorithms.

3) We adopt a queue network to verify the influence of service dependency on the network latency of network applications. It provides a foundation for consolidatin service components according to their dependency. 4) We conduct exhaustive experiments in real-world data centers. We deploy five applications on 290 VMs. The experimental results show that CloudScout can effectively discover service dependency and help to reduce the network latency of network applications and distributed applications.

4. Problem Statement

Service components for enterprise applications are deployed in many hosts and network devices. Service components consolidation is carried out with the consideration of resource constraints, service dependency and network structure. CloudScout is a non-intrusive approach applied for automatically discovers dependent service components. The correlation among service components is analyzed with the time-series information from system monitoring logs. The log mining approach is used for the service dependency discovery with generality and privacy protection features. The iEntropy method is applied to dynamically determine the weight of each resource metric in the service distance calculation. The hierarchical and iterative k-means (HiKM) algorithm is adapted for the dependent service clustering process. The Queue network and network latency optimization are used in the VM consolidation process. Limited accuracy is identified in service dependency discovery and service consolidation process. Sequential dependencies are not focused in the analysis. Complimentary dependency relations are not focused. Resource conflict and availability constraints are not focused in the service consolidation process.

5. Multi Constrained Virtual Machine Allocation Scheme

The cloud services are used to perform the computational tasks in the cloud computing environment. The services are connected with the data centers. The web applications can be build with the cloud services. The service dependency analysis is carried out to discover the similar service components that are belonging to the same web application. The service components can be executed based on the user requests. The services are monitored with throughput and response time parameters. The similarity measures are used to identify the service relationship levels. The clustering process is used to group the service details. The system combines the hierarchical clustering and partitioned clustering methods. The hierarchical clustering is used for the initial centroid estimation process. The K-means clustering scheme is used for the iteratative operations. The services are assigned with reference to the dependency details. The CloudScout scheme is enhanced to support optimal service dependency discovery and service consolidation process. The service dependency discovery process is improved with sequential dependencies and complimentary dependencies. The Multi Objective Service Consolidation (MOSC) scheme is build to allocate the Virtual Machines for the service components. The service consolidation process is upgraded with resource level constraints. The application response time is recued in the system. The service dependency analysis process uses the multi dimensional relationship model. Resource constraint based service consolidation mechanism. Data transmission cost and communication cost are minimized.

The service dependency discovery and service consolidation operations are carried out with log mining mechanism. Different relationship models are used in the service dependency discovery process. The service consolidation process is build with current resource status levels. The system is divided into six major modules. They are System monitoring process, Service dependency classification, Service distance estimation, Clustering process , VM consolidation process and Multi Objective Service Consolidation (MOSC) scheme.

The system monitoring module collects the service and resource usage details. The service dependency classification is used to categorize the services. Service components relationships are analyzed in the distance estimation module. The clustering process is build to group the similar service components. The virtual machine mapping operations are carried out under the VM consolidation process. The Multi Objective Service Consolidation (MOSC) scheme is constructed resource constraint based VM mapping process.

The network and resource usage data values are collected under the system monitoring process. TCP and UDP connection details are collected to identify the connection number details.The CPU, memory, I/O and network resource details are collected as resource usage metrics. The system monitoring details are maintained under the history logs. The service components and dependency categories are identified from the service dependency classification process. Service components are categorized for the correlation analysis process. Service dependencies are categorized with its operating flows. General, sequential and complimentary dependencies are used in the system.

The service distance estimation process is applied to estimate the relationship between the service components. The preprocess is performed to select the features from the system monitoring data values. The iEntropy method is used to assign the weights for the resource metrics. The distance estimation is performed between the pair of service components. The similar service components are grouped in the clustering process. Hierarchical and iterative k-means (HiKM) algorithm is used for the service component clustering process. The initial centroids are estimated using the hierarchical clustering results. The iterative clustering operations are carried out with the selected centroids.

The VM consolidation process is carried out to allocate the virtual machines to the service components. The queue network and network latencies are analyzed for the VM mapping process. The mapping process is performed using the service dependency information. Transmission cost and data communication cost parameters are also used in the mapping process. The Multi Objective Service Consolidation (MOSC) scheme is build to allocate the VMs for the service components with multiple criteria. The VM mapping process is build with parameter selection based model. Resource conflict and availability parameters are used in the VM allocation process. The service consolidation process is performed with different service dependency categories.

6. Conclusion and Future Work

Enterprise applications are build with independently deployable service components provided under the cloud environment. CloudScout approach is employed to discover the dependent service components and service consolidation process. The service dependency discovery process is improved with sequential dependency and complimentary dependency relationships. The service consolidation scheme is enhanced with multi objective functions and resource constraints. The service dependency analysis is used in the virtual machine allocation process. The service dependency analysis is performed with general, sequential and complimentary dependencies. The service consolidation is achieved with current resource status information. Network latency and resource cost are minimized in the system. The system can be enhanced to support commercial cloud operations. The streaming service operations can be interated with the resource allocation mechanism.

References

[1] X. Pu, L. Liu, Y. Mei, S. Sivathanu, Y. Koh, and C. Pu, “Understanding Performance Interference of I/O Workload in Virtualized Cloud Environments,” Proc. IEEE Int’l Conf. on Cloud Computing (CLOUD ‘10), pp. 51-58, Jul. 2010.

[2] Xing Pu, Ling Liu , “Who is your neighbor Net io performance interference in virtualized clouds”, IEEE Transactions On Services Computing, Manuscript Id 2013

[3] K. K. Ram, J. R. Santos, and Y. Turner, "Redesigning Xen's Memory Sharing Mechanism for Safe and Efficient I/O Virtualization," Proc. the 2nd Int’l Workshop on I/O Virtualization (WIOV '10), pp. 2-9, Mar. 2010.

[4] Bora Karaoglu and Wendi Heinzelman, “Cooperative Load Balancing and Dynamic Channel Allocation for Cluster-Based Mobile Ad Hoc Networks”, IEEE Transactions On Mobile Computing, Vol. 14, No. 5, May 2015

[5] J. R. Santos, Y. Turner, G. Janakiraman, and I. Pratt. "Bridging the Gap between Software and Hardware Techniques for I/O Virtualization," Proc. USENIX Annual Technical Conference (ATC'08), pp. 29-42, Jun. 2008.

[6] K. K. Ram, J. R. Santos, Y. Turner, A. L. Cox, and S. Rixner, "Achieving 10 Gb/s Using Safe and Transparent Network Interface Virtualization," Proc. ACM/SIGPLAN/SIGOPS Conf. on Virtual Execution Environments (VEE'09), pp. 61–70, Mar. 2009.

[7] The XenTM Virtual Machine Moniter: http://www.xen.org, 2010.

[8] CitrixR XenServer: http://www.citrix.com, 2011.

[9] Israel Martin-Escalona, Fabio Perrone, Enrica Zola and Francisco Barcelo-Arroyo, “Impact of unreliable positioning in location-based routing protocols for MANETs”, 13th International Wireless Communications and Mobile Computing Conference, IEEE, 2017.