http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (22): 6676-6680, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.6676.6680

Service Components-based Resource Allocation in Services Cloud

Fang Gao, Chuanchang Liu and Yahui Zhang
State Key Lab of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, 100876, Beijing, China

Abstract: As cloud-based services become more numerous and dynamic, service providers have to address the problem on how to allocate the cloud resources to their services. Only by solving this problem can they enjoy the high efficiency and low cost from the cloud. Many researchers focus on resource provisioning based on virtual machines while we propose an allocation strategy based on service components. In this study, the allocation problem is modelled as a bin packing problem. We propose an algorithm and implement the resource allocation system based on it. We deploy several services on our cloud platform for experiments and the results indicate that the allocation system performs well.

Key words: Cloud computing, resource allocation, services cloud, bin packing problem

INTRODUCTION

Cloud computing is a large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platform and services are delivered on demand to external customers over the Internet (Foster *et al*., 2008). It greatly changes the way for IT companies to deliver their softwares/hardwares, also affects how customers will store and access information over the Internet. More and more companies, e.g. Google and Amazon, are offering cloud computing services such as Google's App Engine and Amazon's Elastic Compute Cloud (EC2) (Ardagna *et al*., 2011).

With the rapid development of Internet and Mobile Internet, a large number of applications take full advantage of Web service technique to bring users entirely new experience. Since, the service demand changes dynamically and drastically, cloud is an ideal platform to support the various services. Therefore, many service providers have decided to deploy their services on cloud platform.

The services providers want to assure the quality of services while minimize the costs. Their profit will be affected by the efficiency of the resource provisioning strategy. Therefore resource provisioning is among the most critical issues in cloud research. The resource allocation can be performed at two levels: At the host level and at the Virtual Machine (VM) level. At the host level, it is possible to specify how much of the overall processing power of each core in a host will be assigned to each VM. At the VM level, the VM assigns a specific amount of the available processing power to the individual application services (Calheiros *et al*., 2011).

A lot of work on implementing a highly scalable services cloud platform has been done inside our laboratory. We have designed an effective management system for this platform comprising the following components: Resource monitoring, load prediction, load balancing, auto-scaling and resource allocation. In this study, we are devoted to propose a resource allocation strategy at the VM level for our services cloud platform. Collaborating with the other components, they together improve the resource utilization and scalability of the platform.

RELATED WORK

As cloud computing receives extensive attention, resource allocation has become a new research hotspot because of its great importance. There exits many outstanding achievements in this field.

Zeng and Xu (2011) propose an energy efficiency resource allocation strategy by modeling the allocation problem as a problem of path construction, improving elitist strategy for ant system to optimize resource allocation scheme. In another research (Wuhib *et al*., 2012) a resource management system is implemented upon OpenStack. The system supports an extensible set of management objectives between which the system can switch at runtime, making the resource allocation keep

Corresponding Author: Fang Gao, State Key Lab of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, 100876, Beijing, China

effective according to different requirements from the customers. Both of the researches focus on the allocation at the host level, trying to maximize the physical resource utilization and minimize the costs.

As for services cloud, resource allocation at the VM level is also very important. A problem obsessing services providers is to know exactly how many and what kinds of VMs will be needed for their services. Some researchers (Chang *et al*., 2010; Lin *et al*., 2010) solve this problem in polynomial time. They use as few resources as possible to reduce costs and overhead for setting up the resources. Unfortunately, neither of them mentions how to deploy their services on the VMs they've already possessed.

Compiere ERP (2010) provides an individual VM for each service. This causes waste of resources since customers may not use complete VM capacity which is reserved to serve their requests. This solution is able to guarantee the service quality but will suffer high infrastructure cost.

Since, our services cloud platform is scalable, the amount and types of VMs will change with time so we need to reallocate the VMs to service components. This requires our allocation solution easy to adjust. Therefore we propose a resource allocation strategy for service components taking this factor into consideration.

ARCHITECTURE OF CLOUD MANAGEMENT SYSTEM

The services cloud platform is supposed to be with high availability and scalability. An effective management system is designed to achieve this goal. The resource allocation system is a component inside this system. It collaborates with the other components to provide the cloud platform with high utilization and scalability. Figure 1 shows the architecture of the services cloud platform and we can see the cloud management system in the middle.

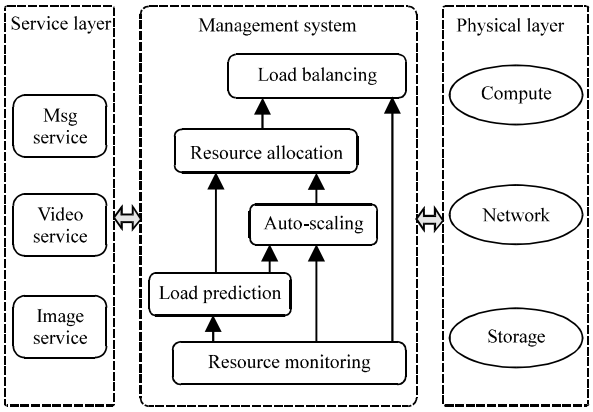

Fig. 1: Architecture of management system

Resource monitoring component: It monitors the cloud platform to retrieve and record the real-time performance metrics like CPU utilization, memory usage and network I/O of physical and virtual resource. It also provides APIs to other components to query the data.

Load prediction component: Based on the data provided by the resource monitoring component, the load prediction component maintains a historical information table recording the prediction value. The prediction reliability in the information table will be updated according to the real load condition.

Auto-scaling component: In this component a scaling model is constructed to make scaling decisions. Based on the real-time performance metrics and the predicted load, the model considers the relative merits of scaling by changing the number of VMs versus changing the size of VMs. Then a combination of horizontal and vertical scaling solution will be performed.

Resource allocation component: The resource allocation is implemented in this component. It will allocate the VMs to different services. The adopted strategy in this component utilizes the results from the load prediction component and the auto-scaling component.

Load balancing component: The load balancing component gets the services deploy solution and the performance metrics of the VMs, dispatches the users' requests to particular VMs according to the load balancing strategy adopted.

MODELLING AND SYSTEM IMPLEMENTATION

Problem description and modeling: According to the description of the cloud management system, the resource allocation component can get the amount of resources that each service needs due to the load prediction component. The auto-scaling component provides the amount of VMs and how many resources each VM holds. Then we adopt an appropriate strategy to deploy the services on the VMs allocated to us. We aim to find a solution easy to scale that is, if we need to reallocate the resources for a specific service, the less VMs be affected the better.

We can model the problem as a variable-sized bin packing problem.

Definition 1: (Classical variable-sized bin packing problem) One is given a list $L = (a1, a2, \ldots, an)$ of items (or elements) and a collection $S = \{S1, S2, \ldots, Sm\}$ of capacities of bins. We have an infinite supply of bins. The capacity of a specific bin is Sk and $Sk \in S$, $1 \leq k \leq m$. A

function s(ai) gives the size of item ai and satisfies $1 \le i \le n$. The problem is to pack the items into a minimum number of bins under the constraint that the sum of the sizes of the items in each bin is no greater than the capacity of the bin.

In this resource allocation case, the supply of bins is limited since we get certain amount of VMs after auto-scaling. We consider the VM as a bin and a particular service as an item to be packed. The computing power of a VM can be regarded as the capacity of a bin and the resources demand quantity of a service can be regarded as the size of an item. Besides, we should pay attention to that the resources of a VM may be not enough for a service. Therefore we need to divide the service into two or more service entities. Our goal is to minimize the capacity waste of all the bins and the number of items inside one bin. We define:

$$x_{ij} = \begin{cases} p, \text{ if i is put in j and occupies capacity p} \\ 0, \text{ otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} p, \text{ if j contains i} \\ 0, \text{ otherwise} \end{cases}$$

Vj: Capacity of bin j, Vj∈S.

Based on the problem description and the definitions above, we construct the model as below:

$$\min \sum_{i=1}^{n} y_{ij}$$

$$\min \sum_{i=1}^{m} y_{ij}$$

$$\text{s.t.} \begin{cases} \sum_{j=1}^{m} x_{ij} = s(a_i) \\ \sum_{i=1}^{n} x_{ij} \le V_j \\ y_{ij} = 0 \text{ or } 1 \\ x_{ij} = 0 \text{ or } p,\ p \le V_j,\ s(a_i) \\ \operatorname{sgn}\left(\sum_{j=1}^{m} y_{ij}\right) = \operatorname{sgn}\left(\sum_{i=1}^{n} x_{ij}\right) = 0 \text{ or } 1 \\ \forall i = 1, 2, \ldots, n;\ \forall j = 1, 2, \ldots, m \end{cases}$$

Algorithm description: First Fit Decreasing (FFD) algorithm is one of the classical algorithms to solve the bin packing problem. Considering the unique requirement in our case, we propose a new algorithm on the base of FFD. Figure 2 shows the processing flow.

First the VMs and services are pre-processed. They are sorted in descending order by the amount of resources the VM can provide or the amount of resources the service needs. Each time we get the first service and the first VM out of the service list and VM list

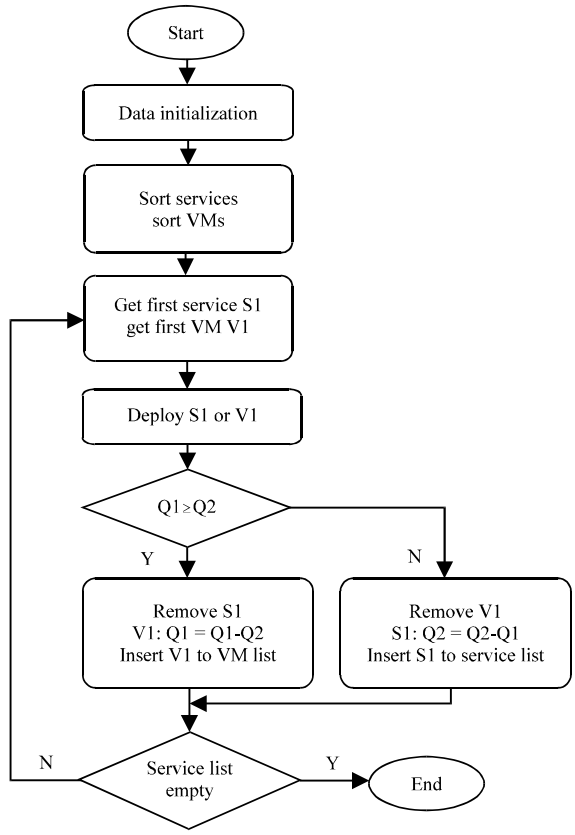

Fig. 2: Algorithm flow

respectively. Then we compare the amount of resources the VM can provide (Q1) with the amount that the service still needs (Q2).

If $Q1 \ge Q2$, the service ought to be deployed on this VM and then removed from the service list. The amount of resources the VM can provide will be updated by Q1-Q2. Finally put the VM back into the VM list according to this value.

If $Q1 < Q2$, the service should be divided into two entities. The first entity should be deployed on this VM. The amount of resources that the other entity needs will be Q2-Q1. Remove the VM from the VM list and put the service back into the service list according to the value Q2-Q1.

Repeat the operations till there is no service in the list.

Implementation of resource allocation: We implement the resource allocation system based on the problem analysis and the algorithm we proposed above. The architecture and the solution are shown in Fig. 3.

Services pool: Here contains all the services waiting to be deployed. Some necessary information such as the

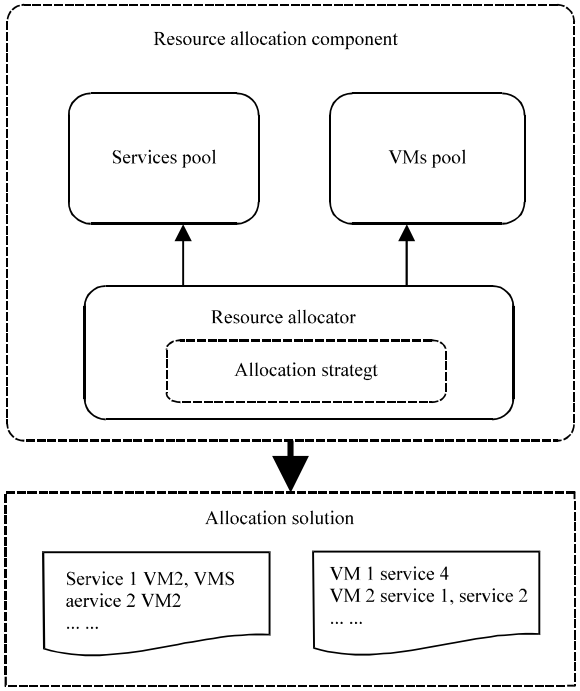

Fig. 3: Architecture and solution

amount of resources the service entity needs is also recorded. The resource allocator will get the services information from this pool.

Vms pool: Similar to the services pool, it contains all the available VMs. The allocator needs to know the location and the remaining resources of a VM in order to make a proper decision.

Resource allocator: This is the core of the resource allocation system. The allocation algorithm we proposed is implemented here. The allocator retrieves necessary information from the services pool and VMs pool, produces a deploy solution conducted by the allocation algorithm.

Allocation solution: In the solution produced by the resource allocator, there are two records. One records all the information where the service is deployed while the other records the services collection each VM supports. These records will be needed by the load balancing component.

EXPERIMENTS

We deploy the services cloud platform in a cluster of four IBM X3650 servers. All the servers share the same configuration. The resource allocation system is implemented as a component inside the cloud

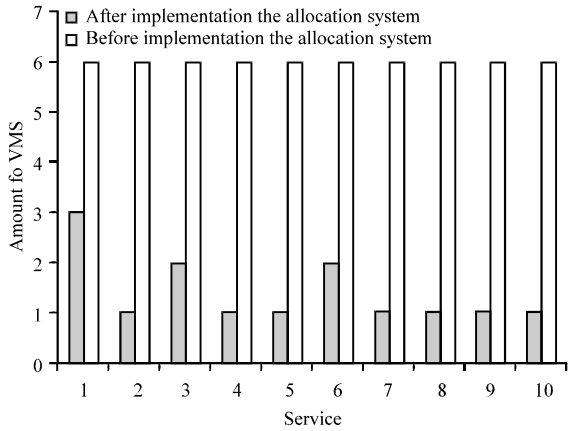

Fig. 4: Comparison of VM amount

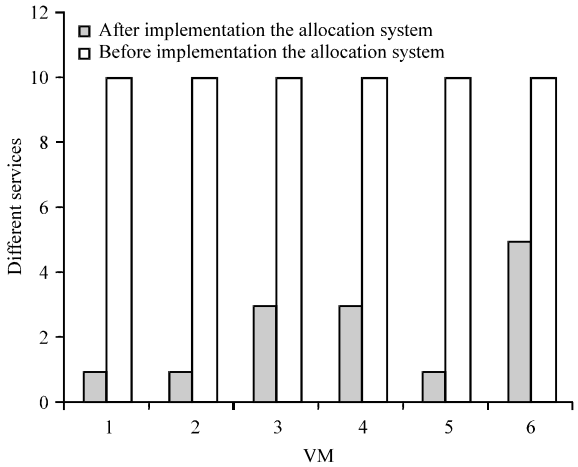

Fig. 5: Comparison of service amount

management system of the services cloud platform. Collaborating with the other components in the management system, they together improve the resource utilization and scalability of the platform. We do some standalone experiments to verify the effectiveness of the resource allocation system itself. The dataset for the experiments is obtained from the resource monitoring component by deploying some real services on the platform. The services are developed inside our laboratory.

Before we implement this allocation system, we use this strategy for service deployment: we deploy each service on every available VM to assure the service quality and high resource utilization. However, this solution is not flexible for scaling. If one of the VMs is shut down or we want to remove or add a service, all VMs will be affected and need reconfiguration. By adopting the new algorithm we propose in Section 4, the problem is solved and we can get a deployment solution easy for adjusting. Figure 4 and 5 show the number of VMs each

service will occupy and the number of different services each VM supports compared to the number before we implement the resource allocation system.

From the figures we can clearly see that the numbers decrease significantly after the allocation system is implemented. This means that fewer VMs need reconfiguration after auto-scaling or the load of service request changing. This indicates that our resource allocation system is able to produce a deployment solution with high flexibility which satisfies our predefined requirements.

CONCLUSION

With the rapid development of cloud computing, more and more service providers have decided to deploy their services on a cloud platform to enjoy the high availability and low cost of cloud computing, but they have to confront the challenging resource allocation problem as well. To address this problem, we construct a mathematic model for the resource allocation problem. We propose an allocation algorithm based on the model and implement a resource allocation system in our services cloud platform. The experimental results indicate that the allocation system works well in the platform. The deploy solution can assure high utilization of resources and is easy for scaling.

ACKNOWLEDGMENT

This study is supported by the National Grand Fundamental Research 973 Program of China under Grant No.2011CB302506 and the Technology Development and Experiment of Innovative Network Architecture under Grant No.CNGI-12-03-007.

REFERENCES

Ardagna, D., B. Panicucci and M. Passacantando, 2011. A game theoretic formulation of the service provisioning problem in cloud systems. Proceedings of the 20th International Conference on World Wide Web, March 28-April 1, 2011, Hyderabad, pp: 177-186.

Calheiros, R.N., R. Ranjan, A. Beloglazov, C.A.F. De Rose and R. Buyya, 2011. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software Pract. Exp., 41: 23-50.

Chang, F., J. Ren and R. Viswanathan, 2010. Optimal resource allocation in clouds. Proceedings of the IEEE 3rd International Conference on Cloud Computing, July 5-10, 2010, Miami, FL., USA., pp: 418-425.

Compiere ERP., 2010. Compiere ERP on cloud. http://www.compiere.com/

Foster, I., Y. Zhao, I. Raicu and S. Lu, 2008. Cloud computing and grid computing 360-degree compared. Proceedings of the Grid Computing Environments Workshop, November 12-16, 2008, Austin, Texas, USA., pp: 1-10.

Lin, W.Y., G.Y. Lin and H.Y. Wei, 2010. Dynamic auction mechanism for cloud resource allocation. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, May 17-20, 2010, Melbourne, Australia, pp: 591-592.

Wuhib, F., R. Stadler and H. Lindgren, 2012. Dynamic resource allocation with management objectives-Implementation for an OpenStack cloud. Proceedings of the 8th International Conference on Network and Service Management, October 22-26, 2012, Las Vegas, NV., USA., pp: 309-315.

Zeng, Z.B. and L. Xu, 2011. Energy efficiency virtual resource allocation strategy for cloud computing. Comput. Syst. Appl., 12: 55-59.