2012 Fourth International Conference on Computational and Information Sciences

A Method for Solving the Performance Isolation Problem in PaaS Based on Forecast and Dynamic Programming

Jiayang Yu
No. 800, Dong Chuan Road
School of Software, Shanghai JiaoTong University
Shanghai, China
E-mail: imcharles@sjtu.edu.cn

Ruonan Rao
No. 800, Dong Chuan Road
School of Software, Shanghai JiaoTong University
Shanghai, China
rao-ruonan@cs.sjtu.edu.cn

***Abstract*—As a Platform as a Service (PaaS) designer, we should satisfy customers' demands of resources and to save resources in a cloud at the same time. In PaaS, one of the main problems is that if one of virtual machines' (VM) workload increases fast in a host, other VMs' performance characteristics may be influenced negatively. To solve the performance isolation problem, this paper gives a method. In this method, we use an algorithm to forecast all VMs' resources usage at first. Then we find which VMs' workloads may increase fast in future. We move these VMs to hosts which have more resources and other VMs' performance characteristics will not be affected at the same time. So the main problem now is which hosts we should use to move these VMs as mentioned above to. In this paper, we use Dynamic Programming method to solve this problem. So, with all of these algorithms, in this paper, we propose a strategy to solve Performance Isolation problem in PaaS.**

Keywords-cloud computing; PaaS; forecast; performance isolation; dynamic programming

I. INTRODUCTION

Cloud computing is now not only a concept in computer science field but also a great business model in many other fields. But nowadays lots of people have got a misconception that resources in a cloud computing environment are infinite or unlimited. In [1], authors remind us that we should avoid this misconception and provide customers the resources as they really need.

A service-level agreement (SLA) is a part of a service contract where the level of service is formally defined. [2] In SLA customer and services provider will make multiple agreements include how many resources services providers may provide to customer. A SLA is a contract. So it is very important to services providers that they should not breach the SLA at any time or they will bear the liability for breach of contract.

As we all know, an application may not use resources in upper bound all the time which is set in SLA. So one of the basic ideas is to allocate more resources to virtual machines (VM) in a real machine host and total of these resources in all VMs are more than the real machine host has. To ensure a services provider won't breach SLA, we should make sure that if one of VMs' workload increases fast in a host, other VMs' performance may not be influenced. This problem is called Performance Isolation problem. To solve this problem, we could move such kind of VMs to other hosts which have more idle resources. As moving a VM to another host may cost lots of time, we should know which VM's workload may increase fast in advance. This forecast needs not to be most accurate because we just want to know the increasing risk or trend of VM's resources demands.

The strategy to move VMs is another problem. If we move a VM with high workload to a host which has fewer resources, the performance isolation problem still exists. If we move the VM to a host which has lots of resources with none other VMs, it's very wasteful. We need find a suitable way to move these VMs.

II. RELATED WORK

In 2005, Chen Guang, et al. made a research in Curve Fitting [3] and give an algorithm. If we got history data of every kind of resource of a VM, we could use this algorithm to get the curve of this VM and then we could forecast the trend of the resource incensement.

In 2010, Chen Xu had a survey of performance management in his master thesis [4], the authors combine the SLA provided for tenants of the largest number of active users to detect the offending tenants, once the monitor found the active users are too many for a long time, then determine the tenant for unauthorized users. Due to the presence of unauthorized users will lead to the performance of other tenants damaged; it will be re-consultation with the tenants and the development of a new SLA.

In 2011, Javier Espadas, et al. had a survey of tenant-based resource allocation model [5]. The authors believe that the VM resources are allocated to tenants depending on the number of active users of the tenant. Therefore, the authors propose a set of active users based multi-tenant virtual machine resource allocation algorithm.

In 2008, Zhi Hu Wang, et al. had a survey of performance evaluation [6]. The authors used x-axis and y-axis. The x-axis includes isolation, security, customization, and scalability. The y-axis includes performance, manageability, and development efficiency. The authors think that a good multi-tenant application should be as much as possible to improve the quality attributes. Allocation of resources should meet the demand for resources for these quality attributes.

978-0-7695-4789-3/12 $26.00 © 2012 IEEE
DOI 10.1109/ICCIS.2012.23

III. FORECAST AND DYNAMIC PROGRAMMING BASED SOLUTION

In Forecast and Dynamic Programming Based Solution (FDPBS), we designed the architecture as the simple class diagram shows in figure 1.

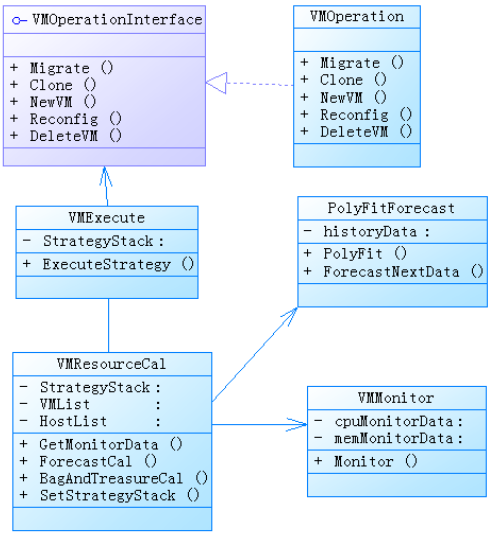

Figure 1. FDPBS Architecture Class Diagram.

As we need to know which VM's resources demand may increase fast, we should monitor every VM in all hosts to get their history monitor data at first. In this paper we take CPU and memory for example. We don't care how to monitor these VMs. What we need is the data. In this paper, we used XML files to save these VMs' monitor data. In this XML file, we record a VM's id, IP address, host, CPU resource upper bound; memory resource upper bound and history monitor data. We don't have to know these data in real time because we just want to know the trend of VMs' resource demands. So it is needless for us to design an Observer-Listener architecture, we just let the VMMonitor class to monitor all the VMs at short intervals. As we all know, operations to VMs may cost lots of time. If we monitor VMs too many times in a short period, last VMs' operations may not finish. By our experience, 10-15 times an hour is proper. It depends on your platform.

The PolyFitForecast class is to calculate the polynomial curve by history monitor data with dichotomy curve fitting algorithm. Then calculate the next possible data in next period by the curve. We cannot say that VMs' resource demands must meet the polynomial formula; in fact the curve is irregular. But nowadays hardware resources such as memories are very cheap. A 32-bit VM may have got 4G memories. For example, the forecast error in 100 Mb is not conclusive and the same to other resources. So let's analysis it, in real environment, it is very rare that VMs' resource demands increase by exponential in a small time period. The logarithm increment is smaller than polynomial increment; to forecast the suddenly increment of resource demands polynomial curve is better.

The VMResourceCal class is to calculate a Strategy Stack (SS) and the VMExcute class is to execute this SS. The stack is to save every VM's operations. VMResourceCal class should tell VMExcute class what to do by this stack. By forecast calculation, we could find which VMs we should move to other hosts according to SLA. Now we should solve the problem that which hosts could hold these VMs. We use dynamic programming to solve the problem. This problem is just like the 0-1 Knapsack problem [7]. We've got lots of hosts. And the idle resources in these hosts are the knapsack bags, the VM which should be move to other hosts are the treasures, and the VMs' resources demands are the treasures' value and weight. The 0-1 Knapsack problem's dynamic programming solution is like below:

```
for vm from 0 to W do
    T[0, vm] := 0
end for

for i from 1 to n do
    for j from 0 to W do
        if   j >= vm[i] then
            T[i, j] := max(T[i-1, j], T[i-1, j-vm[i]] + vm[i])
        else
            T[i, j] := T[i-1, j]
        end if
    end for
end for
```

The input is value (weight) vm for VMs 1 to n; number of distinctVMs n; host knapsack capacity W. Each W in this algorithm represents to one host. And the algorithm of VM Migration Strategy is like below:

```
    for host from 0 to m do
            Calculate host's idle resources
            set the host to BagSet
    end for

    Sort(BagSet)

    for vm from 0 to n do
            PolyFitForecast(vm)
            if (vm has to be moved)
                  set the vm to TreasureSet
            end if
    end for

    for i=0 to from BagSet.size()
        SS.put(Knapsack(TreasureSet, host[i], n))
        TreasureSet.delete(vm.moved)
    end for
```

At first we calculate all the hosts' idle resources to get the BagSet. We sort the set in order to get the biggest bag in the set every time. And then we calculate the VMs' forecast

data as mentioned above to get the TreasureSet. At last we use Knapsack Algorithm to get the SS.

VMExcute class will execute this SS to migrate VMs from former hosts to target hosts.

IV. EXPERIMENT

In this paper, we use VMware vSphere Hypervisor (ESXi) [8] to build a private cloud, use VMware vCenter Server [9] to manage the environment and use Openfiler [10] to build a storage area network (SAN) for all VMs in this private cloud. The SAN is built in a host with 2T storage.

We use other two hosts to hold the VMs and the detail system parameter of them are showed below:

TABLE I. HOSTS' DETAIL SYSTEM PARAMETER

Host name	CPU (Model/Speed)	Memory	Storage
HostA	2* Intel Core 2 Duo E8400 /6144 MHz	2 GB	Use SAN
HostB	2* Intel Core 2 Duo E8400 /6144 MHz	2 GB	Use SAN

We create three VMs in HostB and one in HostA. Each VM we allocate one CPU which speed is 3 GHz, allocate 1 GB Memory and 10 GB storage from SAN. Then we install Apache Hadoop [11] in all these VMs. As we all know, a big Hadoop job especially in its mapping jobs may cost lots of resources in a cluster, so we used an example program called WordCount to test our system. We prepared a big text file which is 512 MB big to test.

In this paper, we only use 'memory' to be the experiment parameter, because it is one of the main parameters of resources and experiments using other resources are same to this experiment.

In HostB, there are three VMs. Each of these VMs is allocated 1 GB memory but there only 2 GB in the host. To ensure the availability of the host, when using these three VMs at the same time, the VMware platform will limit each VM's resources. It is clearly that we should prove that after the system moves a VM from HostB to HostA the total performance becomes better and the actions of the System is necessary. In Figure 2 (and same to Figure 3 and Figure 4), the unit of y-axis is MB and the unit of x-axis is percent of mapping job progress.

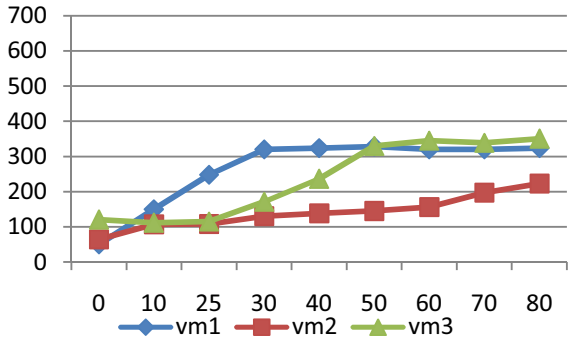

Figure 2. Scene 1, Memory Usage of Three VMs without Migration & without Forecast.

We designed three scenes in the experiment. In scene 1, we don't forecast the memory usage of each VM and don't move any VM. In scene 2, we don't forecast the memory usage of each VM but monitor them. If memory is not enough we will move the VM to other host with idle resources. In scene 3, we will forecast the memory usage of each VM and move the VM to other host in advance.

As we can see in Figure 2, when the WordCount job is running, the highest memory usage is about 350 MB. All of three VMs' memories are limited by the platform. Each VM's memory doesn't reach the upper bound setting by user. And in Figure 5, we can see the total time cost in the WordCount job is about 19 minutes.

In scene 2, we delete the forecast model in our system and just calculate if the resources are not enough and use the dynamic programming to move the VM. In Figure 3, when mapping job progress is to 40%, the biggest memory usage is about 350 MB, the system decide to move a VM to HostA. Then as we can see, all of these VMs' memories allocations are increased. The highest memory usage is about 600 MB. In Figure 5, we can see the total time cost in the WordCount job is about 13 minutes. All of these VMs' performances are protected after 40% mapping job.

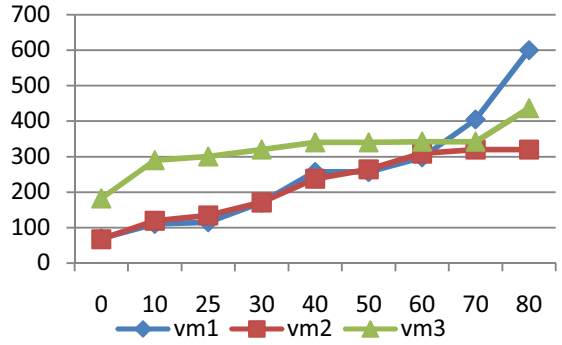

Figure 3. Scene 2, Memory Usage of Three VMs with Migration & without Forecast.

In Scene 3, we use the forecast algorithm to forecast the usage of memories of all VMs and calculate if the resources are not enough in advance. As we can see in Figure 4, when the mapping job progress is to 30%, the system forecast the usage of memories may become to about 350 MB, which means the VMs' memories may be limited by the platform. So the system decides to move a VM to HostA. Then we can see that the increasing rate of memories usage is never limited then. The highest memory usage is about 800 MB. In Figure 5, we can see the total time cost in the WordCount job is about 10 minutes. All of these VMs' performances are protected during the whole mapping job.

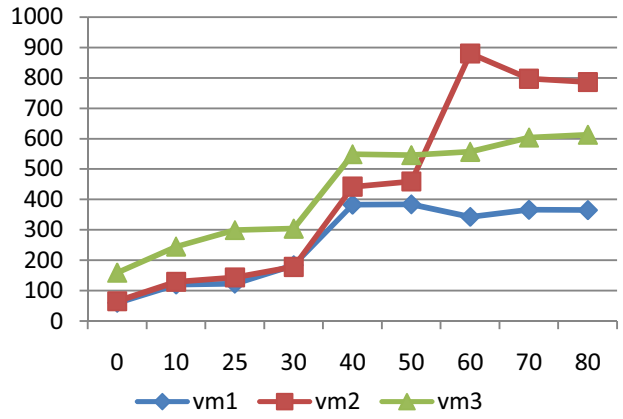

Figure 4. Scene 3, Memory Usage of Three VMs with Migration & with Forecast.

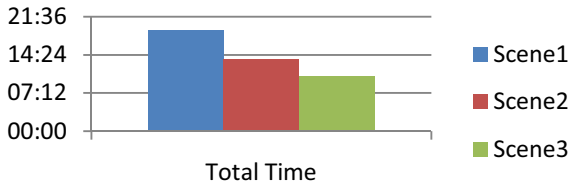

Figure 5. Word Count Cost Total Time in these Three Scenes.

As is mentioned above, the total performance of the WordCount job in scene 3 is better than that in scene 2. And the total performance of the WordCount job in scene 2 is better than that in scene 1. That means in PaaS, performance isolation problem is very important. If we don't care about this problem the performance of VMs may be very low. VM Migration is important then; we can move a VM which need lots of resources to other hosts. After migration, the performances of VMs are ensured. Forecast work is very useful. Although resource forecast may not be most accurate in real environment, it may help us to find the risk of resource shortage in advance. That means we could avoid the performance risks. Dynamic programming helps us to find an efficient way to place all VMs should be moved.

V. CONCLUTION

We design and implement the method of solving performance isolation problem in PaaS. We use curve-fitting forecast method to calculate resources usage of VMs in advance and determine which VMs need to migrate. We use dynamic programming method to decide a best strategy of VM migrating. We design and finish an experiment to prove the method of solving the performance isolation problem in PaaS is right.

Next work, we will design a better math model of curve-fitting to get a more accurate forecast result. We will design a better method of resources calculating to cover more scenes in PaaS.

REFERENCES

[1] Michael Armbrust, et al. Above the Clouds: A Berkeley View of Cloud Computing, Electrical Engineering and Computer Sciences, Technical Report No.UCB/EECS-2009-28, University of California at Berkeley, 2009.

[2] Service-level agreement. 2012. http://en.wikipedia.org/wiki/Service-level_agreement.

[3] CHEN Guang, REN Zhi-liang, SUN Hai-zhu. Curve Fitting in Least-Square Method and Its Realization with Matlab. 2005.

[4] Chen Xu. Research on Performance Management Mechanism Based On SLA in SaaS Application. 2010.

[5] Javier Espadas, et al. A Tenant-based Resource Allocation Model for Scaling Software-as-a-Service Applications over Cloud Computing Infrastructures. Future Generation Computer Systems, Accepted 24 October 2011.

[6] Zhi Hu Wang, et al. A Study and Performance Evaluation of the Multi-Tenant Data Tier Design Patterns for Service Oriented Computing. IEEE International Conference on e-Business Engineering. 2008.

[7] Knapsack problem. 2012. http://en.wikipedia.org/wiki/Knapsack_problem

[8] VMware vSphere Hypervisior, http://www.vmware.com/cn/products/datacenter-virtualization/vsphere-hypervisor/overview.

[9] VMware vCenter Server, http://www.vmware.com/cn/products/datacenter-virtualization/vcenter-server/overview.

[10] Openfiler, http://www.openfiler.com .

[11] Apache Hadoop, http://hadoop.apache.org .