Active Monitoring: ERT Based Load Balancing in Cloud

Priyanka Singh* and Shubham Pandey*

*Department of CSE, Department of Information Technology, SRM University, Kattankulathur-603023, INDIA

Accepted 11 April 2014, Available online 15 April 2014, Vol.4, No.2 (April 2014)

Abstract

In this paper we have proposed and implemented an active monitoring method for Cloud Infrastructure. Cloud computing environment is a model for delivering information technology services in which resources are retrieved from the internet through web-based tools and applications, rather than a direct connection to a server. Users can set up and boot the required resources and they have to pay only for the required resources. Thus, in the future providing a mechanism for efficient resource management and assignment will be an important objective of Cloud computing. In this project we propose an Expected Response time (ERT) based load balancing and dynamic scheduling approach that allocate resources based on the load of Virtual Machines (VMs) on Infrastructure as a service (IaaS). This method enables users to dynamically add and/or delete one or more instances on the basis of the load and the conditions specified by the user. Our objective is to develop an effective load balancing algorithm using Virtual Machine Monitoring to maximize or minimize different performance parameters (throughput for example) for the Clouds of different sizes (virtual topology de-pending on the application requirement).

Keywords: Load Balancing, Virtualization, IaaS Cloud, Cloud Computing, ERT, Resource Allocation, Distributed Processing.

1. Introduction

In this paper we have implemented a cloud application platform which enable the whole infrastructure having both virtual and physical components to appear as a single, highly extensible and reliable system, which has the ability to operate itself and adaptable to the failures and changing load (A. Haider et al, 2009). It can also be provisioned either by an application or simply by a web request (Rochwerger B. et al, 2011). Cloud computing (http://en.wikipedia.org/wiki/Cloud_computing) operates in three major layers- Cloud infrastructure (commonly known as Infrastructure as a service or IaaS), cloud application platform (commonly known as platform as a service or PaaS) and cloud application (commonly known as Software as a service or SaaS). We have proposed a cloud service model (Eric A. et al, 2010), here that is encapsulated using several layers as:

1.1 Cloud Application Layer: As a part of Software as service it provides various applications and policies to cloud consumers.

1.2 Cloud Business Layer: it provide the vision for various business objective, cloud enabled business application, software, policies for business deployment and various associated analysis framework.

1.3 Cloud Platform Layer: as a part of cloud application platform (PaaS) it abstract a middleware between cloud consumer and cloud platform for offering various technical solutions, application to consumers based on their demand.

1.4 Cloud Operating System Layer: as a part of cloud application platform it provide a model for application virtualization, cloud software and hardware provisioning, billing, task flow execution load balancing and monitoring for various event in cloud environment.

1.5 Virtualization Layer: as a part of Infrastructure as a service virtualization layer provide a core virtualize
infrastructure and potentially useful foundation for cloud computing to offer service on-demand.

1.6 Cloud Physical layer: as a part of Infrastructure as a service cloud physical layer will provide various physical computing component, storage and network resources and various measurement tools for the security that are virtualized and cloud enabled to support cloud environment.

By taking all these factors we have implemented an expected response time based approach for analysing and monitoring the behaviour of the virtual machines at various workloads. We are using this algorithm to distribute our workload across several virtual machine to avoid the performance bottleneck and to increase the throughput.

2. Proposed Model

2.1 Existing VM Load Balancer

Virtual machine enables the abstraction of an Operating System and Application running on it from the hardware (OpSource Cloud). The interior hardware infrastructure services interrelated to the Clouds are modeled in the simulator by a Datacentre element for handling service requests. These requests are application elements sandboxed within VMs, which need to be allocated a share of processing power on Datacentre’s host components (Endo P. et al, 2011). Datacentres object manages the datacentre management activities such as VM creation and destruction and does the routing of user requests received from User Bases via the Internet to the VMs (Amazon Virtual Private Cloud). The Data Centre Controller, uses a VM Load Balancer to determine which VM should be assigned the next request for processing.

2.2 Proposed VM Load Balancing Algorithm

To provide the higher degree of satisfaction in the form of throughput and CPU Utilization we have proposed and implemented an approach for load balancing based upon the expected response time of VM. By using this approach we are distributing our workload across several VM for a successful task flow execution as shown in Figure 2.

2.3 Expected Response Time (ERT) based Load Balancing Algorithm

The Proposed Expected Response Time (ERT) based Load balancing algorithm is divided into three parts.

The first phase is the initialization phase. In the first phase, the expected response time of each VM is to be find. In second phase find the efficient VM, in Last Phase return the ID of efficient VM.

The working of algorithm is shown using the Flow Chart shown in figure 2.

![Flowchart for ERT based Load Balancing Algorithm](image)

Fig. 3. Flowchart for ERT based Load Balancing Algorithm.

To distribute the load across several nodes (VMs) we have presented the algorithm as shown in flow diagram represented in figure 3. We are assuming the threshold as a fraction from 0 to 100 where we are taking the default value as 10. Because we are assuming that none of the virtual machine will be idle after initialization. The steps for the computation to achieve the load balance are as follows. We are taking some assumption as:

{Some assumptions}
\( \text{VM}_i = i^{th} \text{ virtual machine} \)

N= Total number of virtual machine

Available set of VM = \{\emptyset\}

Satisfactory set of VM = \{\emptyset\}

Algorithm 1: ERT Based Load Balancing

1. In the initialization phase we compute whether the VM is available or not.
   1.1. If the machine is available then add it to the set of available VMs. And GOTO step 2.
   1.2. Else wait for random amount of time \( k \) and GOTO step 1.1.
2. For each VM in set of available VMs estimate the load on each machine.
   2.1. Now for each VM compare the load with the threshold. As we already have discussed that the default threshold is 10.
   2.2. If load (\( \text{VM}_i \)) < Threshold then add the \( \text{VM}_i \) to the satisfactory set of VM.
   2.3. Else GOTO 1.2
3. Now for each \( \{ \text{VM}_1, \text{VM}_2, \ldots, \text{VM}_P \} \) in the satisfactory set of VM find the expected response time.
4. Now find the efficient VM based upon the constraints given by the user (based upon user request) and the corresponding ERT.
5. Return the ID of the VM.

2.4 Algorithm to calculate Expected Response Time

In this approach based upon the prior response time estimated from the previous requests we are analyzing the ERT for specific VM. Concept of supervised Learning is used in this estimation to incorporate more reliable outcomes.

Algorithm 2: Expected Response Time (ERT)

Let \( T_r \) be the actual response time for the previous request, \( N \) be the total number of VM and \( W_i \) represent load on the \( \text{VM}_i \) (ith virtual machine). Then for \( n \)th number of request from the consumer the expected response time \( \mu(t) \) can be expressed as

\[
\mu(t) = \frac{N.T_r}{\sum_{i=1}^{N} W_i}
\]

Where we are assuming that at any time on the ith node \( W_i \neq 0 \).

Modules

The modules that we have implemented are:

3.1 Bucket Creation: Browser allows to easily create Buckets in all regions supported by commercial cloud. Once created a new bucket, consumer can create virtual folders to organize files, and upload and download files to

and from commercial cloud database as shown in Figure 4.

Fig. 4. Bucket Creation Module

3.2 Uploading Instances: VM import/export enables to easily import virtual machine image from existing environment to commercial cloud instances and export them back to on-premises environment as shown in figure 5 and 6. This offering allows to leverage an existing investment in the virtual machine that built to meet IT security. Configuration management and compliance requirements by seamlessly bringing those virtual machines into commercial cloud as ready to use instances. Consumer can easily export imported instances back to an on-premises virtualization infrastructure, allowing to deploy workload across IT infrastructure.

Fig 5. Instance Uploading Module

Fig 6. Instance Downloading Module
3.3 Monitoring Instances: ERT based load balancer provides a reliable, scalable, and flexible monitoring solution that can start using within minutes. No longer need to set up, manage, or scale own monitoring systems and infrastructure. Using this approach admin can easily monitor the instances. It also lets admin to programmatically retrieve consumer instances. It has the features for monitoring data, view graphs, and set alarms to help you troubleshoot, spot trends, and take automated action based on the state of Cloud environment. The Monitoring Module is coded using Java language which is such as:

```java
public class monitoring extends javax.swing.JFrame {
    public monitoring() {
        try {
            Class.forName(com.mysql.jdbc.Driver);
            Connection con = DriverManager.getConnection(jdbc:mysql://localhost:3306/loadbalancing, root, root);
            st = con.createStatement();
        } catch (Exception e) {
            e.printStackTrace();
        }
        initComponents();
        }

        private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
            try {
                FileInputStream fis = null;
                String filebody=jTextArea2.getText();
                String filename=jTextField1.getText();
                FileWriter fstream = new FileWriter(proxyserver+filename);
                BufferedWriter out = new BufferedWriter(fstream);
                out.write(filebody);
                File file = new File(jTextField1.getText());
                fis = new FileInputStream(jTextArea2.getText());
                String str=jTextArea2.getText();
                st.executeUpdate(insert into upload values('admin','admin','+filename+','+filebody+','fs','time'));
                FileOutputStream fos = new FileOutputStram(F:\+filename);
                fos.write(str);
                fos.close();
                out.close();
                JOptionPane.showMessageDialog(this, File Has Been Uploaded Successfully);
            } catch (Exception ex) {
                Logger.getLogger(upload.class.getName()).log(Level.SEVERE, null, ex);
            }
        }

        private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
            proxyserver pr=new proxyserver();
            pr.setVisible(true);
        }

        private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
            clientevent ce = null;
            try {
                ce = new clientevent();
            } catch (Exception ex) {
                Logger.getLogger(monitring.class.getName()).log(Level.SEVERE, null, ex);
            }
        }

        In this way using this module Admin can monitor log event as well as characteristic of virtual machines on several instances as shown in Figure 7.

        Fig 7. Monitoring Instance Module

        The administrator has the ability to check the availability of virtual machines using ERT based approach as shown in Figure 8.

        Fig 8. Monitoring Availability of VM
```
Cloud administrator can also monitor the client log event. And the instances imported and exported on the cloud as shown in figure 9.

Fig 9. Monitoring Client Events

In this way cloud administrator has the ability to monitor and view load statistic as shown in Figure 10 and 11.

Fig 10. Weighted Active Monitoring

Fig 11. Monitoring the ERT and efficiency of each VM for request

3.4 Performance: The ERT based load balancing approach provide an efficient mechanism to balance the load across several machine. In our experiment we have monitor the performance of virtual machines based upon simultaneous requests as shown in Figure 12. Which leads to the conclusion that this approach provide an optimal solution for the load balancing.

Fig 12. For each request performance of VM

Conclusion

In this paper we have proposed and implemented an ERT based load balancing concept through which the request from the cloud consumer can be handled efficiently and for the simultaneous requests load can be distributed so that to minimize the overhead and to avoid the performance bottleneck at any node. This approach also provide active monitoring for the clients events .admin can monitor all the instances imported and exported by the consumers and also can monitor event log for several consumers at each virtual machine. At last we are analyzing the performance of the virtual machines in terms of efficiency and throughput based upon different load at several time slots. Our future work is to implement ERT based algorithm in hybrid cloud environment. and to monitor the performance of ERT based algorithm in hybrid cloud.

References


