International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 2, February- 2013

Proposed Efficient and Enhanced Algorithm in Cloud Computing

Tejinder Sharma
Amritsar College of Engineering and Technology

Dr.Vijay Kumar Banga
Amritsar College of Engineering and Technology

Abstract

Cloud computing, as of today has boomed and enhanced the use of internet as well as network services where the capability of one node can be used by other nodes. Gigantic-scale heterogeneous distributed computing environments (like Clouds and Computational Grids) can easily access the vast amount of computing resources at very economical cost. This has implications for many technical issues in Service Oriented Architectures and Internet of Services (IoS)-type applications; including high interoperability, fault tolerance, high availability and scalability. Central to these issues is the establishment of effective load balancing techniques. It is clear that the scale and complexity of these systems makes centralized assignment of jobs to specific servers infeasible; requiring an effective distributed solution. To properly manage the resources of the service provider, we require balancing the load of the jobs that are submitted to the service provider.

1. Introduction

Cloud computing enables shared servers to provide resources, software and data for collaborative services on demand with high interoperability and scalability [2]. Today, there are more than a hundred million computing devices connected to the Internet and many of them are using cloud computing services daily. According to the IDC's anticipation , the SaaS (Software As A Service) market reached $13.1 billion in revenue at 2009 will grow to $40.5 billion by 2014 at a compound annual growth rate(CAGR) of 25.3%.[2] These networked devices submit their requests to a service provider and receive the results back in a timely manner without the involvement of the service complexity related to information storage and process, interoperating protocols, service composition, communications and distributed computation, which are all relied on the network and the backend servers to offer desirable performance. However, there are a number of technical challenges that need to be tackled before these benefits can be fully realized, which include system reliability, resource provisioning, and efficient resources consuming etc[2].

Among them, load-balancing is a necessary mechanism to increase the service level agreement (SLA) and better uses of the resources. It is a process of reassigning the total load to the individual nodes of the collective system to make resource utilization effective and to improve the response time of the job, simultaneously removing a condition in which some of the nodes are over loaded while some others are under loaded. The concept of load balancing is not typical to Cloud platforms and has been around for a long time in the field of distributed systems. In its most abstract form, the problem of load balancing is defined by considering a number of parallel machines and a number of independent tasks, each having its own load and duration.

A load balancing algorithm which is dynamic in nature does not consider the previous state or behaviour of the system, that is, it depends on the present behaviour of the system. The important things to consider while developing such algorithm are : estimation of load, comparison of load, stability of different system, performance of system, interaction between the nodes, nature of work to be transferred, selecting of nodes and many other ones. The load considered here can be in terms of CPU load, amount of memory used, delay or network load. The main goal of the load balancing in the cloud computing is to

assign the tasks to the machines, therefore increasing their load, in such a way as to optimize their utility. Depending on the source of the tasks, the load balancing problem can be classified as: online load balancing where the set of tasks is known in advance and cannot be modified and online load balancing in the situation that the task set is not known in advance and tasks arrive in the system at arbitrary moments of time. In the case of Cloud computing we can consider load balancing at two different levels:

Cloud provider level and Cloud client level[1].

From the point of view of the Cloud provider, the load balancing problem is of type online and is mapped in the following way:

- The parallel machines are represented by the physical machines of the Cloud's clusters
- The tasks are represented by client requests for virtual resources
- Cloud client requests can arrive at arbitrary moments of time

In the case of Cloud client's virtual platform, the load balancing problem is mapped in the following way:

- The parallel machines translate into the virtual resources that the Cloud client has currently running
- The tasks translate into client requests to the Cloud client's platform
- End user requests can arrive at arbitrary moments of time

As a consequence, in Cloud platforms, load balancing is an online problem where end user requests that enter the Cloud client's application need to be distributed across the Cloud client's instantiated virtual resources with the goal of balancing virtual machine load or minimizing the number of used virtual machines.

In this paper we focus on server-based load balancing solution and try to minimize the service response times and hence data transfer cost by redirecting requests to an optimal remote server that is chosen in terms of communication latency and workload. In this way we can reduce the capital and operational cost of the client

2. Algorithms used to distribute the work load to the client

A. Round Robin Algorithm:

It is the simplest algorithm that uses the concept of time quantum or slices Here the time is divided into multiple slices and each node is given a particular time quantum or time interval and in this quantum the node will perform its operations. The resources of the service provider are provided to the client on the basis of this time quantum. It selects the load on random basis and leads to the situation where some nodes are heavily loaded and some are lightly loaded. Though the algorithm is very simple but there is an additional load on the scheduler to decide the size of quantum[8].

B. Equally Spread Current Execution Algorithm:

In spread spectrum technique load balancer makes effort to preserve equal load to all the virtual machines connected with the data centre. Load balancer maintains an index table of Virtual machines as well as number of requests currently assigned to the Virtual Machine (VM). If the request comes from the data centre to allocate the new VM, it scans the index table for least loaded VM. In case there are more than one VM is found than first identified VM is selected for handling the request of the client/node, the load balancer also returns the VM id to the data centre controller. The data centre communicates the request to the VM identified by that id. The data centre revises the index table by increasing the allocation count of identified VM. When VM completes the assigned task, a request is communicated to data centre which is further notified by the load balancer. The load balancer again revises the index table by decreasing the allocation count for identified VM by one.

C. Throttled Load balancing Algorithm:

In this algorithm the load balancer maintains an index table of virtual machines as well as their states (Available or Busy). The client/server first makes a

request to data centre to find a suitable virtual machine (VM) to perform the recommended job. The data centre queries the load balancer for allocation of the VM. The load balancer scans the index table from top until the first available VM is found or the index table is scanned fully. If the VM is found, the load data centre. The data centre communicates the request to the VM identified by the id. Further, the data centre acknowledges the load balancer of the new allocation and the data centre revises the index table accordingly. While processing the request of client, if appropriate VM is not found, the load balancer returns -1 to the data centre. The data centre queues the request with it. When the VM completes the allocated task, a request is acknowledged to data centre, which is further apprised to load balancer to de- allocate the same VM whose id is already communicated.

3. Proposed Work

The efficient and enhanced algorithm

Step 1: Find the Vm from the list of vm whose current allocation count is less than max allocation of VM list.
Step2: Return the id of Vm.
Step 3 : Check the status of Vm BUSY/AVAILABLE from vmid.
Step 4: If AVAILABLE, count the current allocations If it is able to handle the client request Then return Vmid and that Vmid is allocated to client.
Step 5: If busy, Then return -1 and wait till we find the next available Vm and request is queued

The purpose algorithm find the expected Response Time of each Virtual Machine because virtual machine are of heterogeneous platform, the expected response time can be find with the help of the following formulas:

$$\text{Response Time} = Fin_t - Arr_t + TDelay \quad (1)$$

Where,Arr_t is the arrival time of user request and Fin_t is the finish time of user request and the transmission delay can be determined using the following formulas

$$TDelay = Tlatency + Ttransfer \ldots (2)$$

Where, TDelay is the transmission delay T latency is the networklatency and T transfer is the time taken to transfer the size of data of a single request (D) from source location to destination.

$$Ttransfer = D / Bwperuser \quad (3)$$

$$Bwperuser = Bwtotal / Nr \quad (4)$$

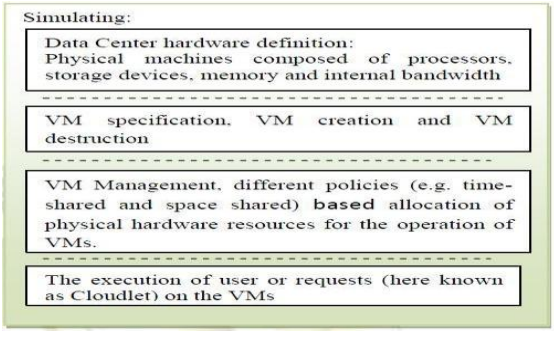
Where, Bwtotal is the total available bandwidth and Nr is the number of user requests currently in transmission. The Internet Characteristics also keeps track of the number of user requests in between two regions for the value of Nr.

4. Simulation and Performance Analysis:

The simulation and Performance Analysis will be done by using the cloud analyst tool

Cloud Analyst[7]:
The deployment of large scale applications is quite economical and easy by using clouds. The cloud also generates the new issues for developers. The various users access the internet applications around the world, and because popularity of applications may vary along the world, so experience in the use of application can also vary. Quantifying impact of different number of simultaneous users, network in applications and geographic location of relevant components is very hard to attain in real testbeds, because of presence of elements which can never be controlled nor be predicted by developers. so, other methodologies must be used that allow the quantification of such parameters. To allow the repeatability and control of experiments, simulator like CloudSim is used. The simulation experiment apply both infrastructures and applications. Therefore, simulation needs the effort from application developer to model both the software in language and the target infrastructure that is interpreted by the simulator. CloudSim is a toolkit used for model ling, experimentation and simulation The Cloud Analyst is a GUI based tool which is developed on CloudSim architecture. Cloud Analyst separates the simulation experimentation exercise from a programming exercise, as modeller should focus on the simulation complexities rather than to spare much time on the technicalities of programming. The Cloud Analyst also facilitates the modeller to operate series of simulation experiments with slight change in respective parameters in easy and quick way.

The main components of cloud analyst are:



GUI Package: It provides graphical user interface to configure the various simulation parameters and act as front end controller for the application.

S**imulation**: This is the most vital component accountable for enduring the simulation parameters, originating and executing the simulation. Based on the various specifications it executes the simulation .

UserBase: The user base is modelled to represent the traffic generated by users who deploy application.

DataCenterController. This component is used to control the various data center activities.

InternetCharacteristics.: In this component various internet characteristics are modelled simulation, which includes the amount of latency and bandwidth need to be assigned between regions, the amount of traffic, and current performance level information for the data centers.

VmLoadBalancer. The responsibility of this component is to allocate the load on various data centers according to the request generated by users. One of the three given policies can be selected. The given p olicies are round robin algorithm, equally spread current execution load, and throttled. By default round robin is used .

CloudAppServiceBroker: The responsibility of this component is to model the service brokers that handle traffic routing between user bases and data centers. The service broker can use one of the routing policies from the given three policies which are closest data center, optimize response time and reconfigure dynamically with load. The closest data center routes the traffic to the closest data center in terms of network latency from the source user base. The reconfigure dynamically with load routing policy works in the sense that whenever the performance of particular data center degrades below a given threshold value then the load of that data center is equally distributed among other data centers. [4][5]

Fig1.2: Cloud Analyst Tool

5. Conclusion:

In this paper we have proposed a new algorithm for load balancing and it will be implemented on cloud analyst an abstract of cloud computing. Proposed algorithm will find the expected response time of each Vm and will send the Id of Vm with least load and status=available to data controller for the allocation of new request.

6. Refrences:

[1]. Eddy Caron , Luis Rodero-Merino "Auto-Scaling , Load Balancing and Monitoring in Commercial and Open-Source Clouds " Research Report ,January2012

[2] Rich Lee,Bingchiang Jeng"Load balancing Tactics in Cloud"2011 International Confrence on Cyber Enabled Distributed Computing and Knowledge Discovery

[3] Wenghong Tian, Yong Zhao, Minixian Xu, Chen Jing"A Dynamic and Integrated load balancing Scheduling Algorithm For Cloud Datacenters",Proc. Of IEEE CCIS 2011

[4]. Bhathiya Wickremasinghe "Cloud Analyst: A Cloud-Sim-based Tool for Modeling and Analysis of Large Scale Cloud Computing Environments. MEDC Project" ,Report 2010 .

[5]. Bhathiya Wickremasinghe ,Roderigo N. Calherios "Cloud Analyst: A Cloud-Sim-based Visual Modeller For Analysing Cloud Computing Environments and Applications". Proc of IEEE International Confrence on Advance Information Networking and Applications,2010.

[6]. R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and Simulation of Scalable Cloud Computing Envir onments and the CloudSim Toolkit: Challenges and Opportunities," Proc. of the 7th High Performance Computing and Simulation Conference (HPCS 09), IEEE Computer Society, June 2009.

[7]. http://www.cloudbus.org/cloudsim.

[8] Tanveer Ahmed,Yogendra Singh "Analytical Study of Load Balancing Techniques Using Cloud Analyst" IJERA ,ISSN:2248-9622,Vol2,Issue2,Mar-Apr2012