Fusion-based Load-aware Resource Allocation on Cloud Infrastructure

GRADUATE PROJECT

Submitted to the Faculty of
the Department of Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, Texas

In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

By

Srinivas Thota
Summer 2017

Committee Members

Dr. Dulal C Kar
Committee Chairperson

Dr. Ajay K Katangur
Committee Member
ABSTRACT

One of the main challenges posed to cloud computing is limited availability of resources for extensive amount of computation required. In the literature, numerous attempts have been made to optimize resource utilization with an objective to improve efficiency of cloud. As the number of requests for cloud services increases, it becomes difficult for the system to balance the load and serve user requests in a stipulated time. Load Balancing is a well known NP-Complete Problem. This project proposes a fusion-based Load aware resource allocation algorithm (FLA) approach that exploits three existing load balancing algorithms: traditional Round-robin algorithm (RR), basic genetic algorithm (BGA) and priority-based genetic algorithm (PGA), with an objective to improve the performance of the system. The idea of fusion lies in considering the variable amount of user requests to the cloud system. The proposed algorithm is intended to use Round-robin algorithm (RR) when there is relatively light load, basic genetic algorithm (BGA) when there is an intermediate load, and priority-based genetic algorithm (PGA) when the system encounters heavy load. Moreover, the goal of this project is also to determine the optimum threshold values for the load to be considered as light, intermediate and heavy. The optimum threshold values will be computed using several parameters such as the number of incoming user requests, CPU, RAM and bandwidth of VMs on the cloud. Utilization of CPU plays a vital role in the VM load. Hence, CPU used by the VM can be considered as a VM load. The simulations will be carried out using CloudSim 3.0.

Keywords: Cloud computing, resources, Load Balancing, Genetic Algorithm.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>ii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vi</td>
</tr>
<tr>
<td>1 BACKGROUND AND RATIONALE</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Cloud Computing Service Models</td>
<td>1</td>
</tr>
<tr>
<td>1.2.1 Software as a service (SaaS)</td>
<td>2</td>
</tr>
<tr>
<td>1.2.2 Platform as a service (PaaS)</td>
<td>2</td>
</tr>
<tr>
<td>1.2.3 Infrastructure as a service (IaaS)</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Cloud Computing Deployment Models</td>
<td>3</td>
</tr>
<tr>
<td>1.3.1 Public Cloud</td>
<td>3</td>
</tr>
<tr>
<td>1.3.2 Private Cloud</td>
<td>3</td>
</tr>
<tr>
<td>1.3.3 Community Cloud</td>
<td>4</td>
</tr>
<tr>
<td>1.3.4 Hybrid Cloud</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Prior Work</td>
<td>4</td>
</tr>
<tr>
<td>1.5 Problem Description</td>
<td>5</td>
</tr>
<tr>
<td>2 NARRATIVE</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Genetic Algorithm</td>
<td>8</td>
</tr>
<tr>
<td>3 SYSTEM DESIGN</td>
<td>10</td>
</tr>
<tr>
<td>3.1 Software and Hardware Requirements</td>
<td>10</td>
</tr>
<tr>
<td>3.2 Simulation Tool</td>
<td>10</td>
</tr>
<tr>
<td>3.3 CloudSim Layered Architecture</td>
<td>11</td>
</tr>
<tr>
<td>3.3.1 The main components of the CloudSim framework</td>
<td>11</td>
</tr>
<tr>
<td>3.4 Proposed System Design</td>
<td>12</td>
</tr>
<tr>
<td>3.5 Proposed System Implementation</td>
<td>13</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.5.1 Methodology</td>
<td>13</td>
</tr>
<tr>
<td>3.5.2 Round-robin Implementation</td>
<td>13</td>
</tr>
<tr>
<td>3.5.3 Basic Genetic algorithm (BGA)</td>
<td>14</td>
</tr>
<tr>
<td>3.5.3.1 Steps in the basic genetic algorithm (BGA)</td>
<td>14</td>
</tr>
<tr>
<td>3.5.4 Priority based Genetic Algorithm (PGA) Implementation</td>
<td>15</td>
</tr>
<tr>
<td>4 EVALUATION AND EXPECTED RESULTS</td>
<td>19</td>
</tr>
<tr>
<td>4.1 Software Requirements</td>
<td>19</td>
</tr>
<tr>
<td>4.2 Experimental Setup</td>
<td>19</td>
</tr>
<tr>
<td>4.3 Expected Results</td>
<td>20</td>
</tr>
<tr>
<td>BIBLIOGRAPHY AND REFERENCES</td>
<td>21</td>
</tr>
</tbody>
</table>
LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>Page</th>
</tr>
</thead>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Basic Genetic Algorithm Structure.</td>
</tr>
<tr>
<td>2</td>
<td>Basic Genetic Algorithm Flow.</td>
</tr>
<tr>
<td>3</td>
<td>Layered CloudSim architecture.</td>
</tr>
<tr>
<td>4</td>
<td>Fusion-based Load aware Resource Allocation on Cloud Infrastructure.</td>
</tr>
<tr>
<td>5</td>
<td>Flow Chart of Priority-based Genetic Algorithm Method.</td>
</tr>
</tbody>
</table>
CHAPTER 1

BACKGROUND AND RATIONALE

1.1 Introduction

For a long period of time, cloud computing has been a very important topic all over the world. It acts as a medium through which internet users are able to store, access, share and process information using browsers or networks. It helps reduce the costs required to set up complex computational systems. Minimum management efforts, independence of the system, scalability and reduced costs are some of the features [11] of a convenient cloud system. As the cloud technology advances, there is an increase in the rate of deployment and adoption of the cloud system to carry out various internet-related activities. It is, therefore, important to evaluate the suitability and performance of cloud technology in different environments. Simulation and modeling are used to test various issues on performance and security in a cloud system. Whenever a software system needs to be tested using the cloud, various tools and techniques must be present in order to solve various concerns.

1.2 Cloud Computing Service Models

Cloud computing, [5] which is popularly known as the cloud is the provision of all the requirements needed for cloud computing starting right from the cloud centers to the users and paying for exactly what one uses. Cloud computing is a self-service facility. It also is flexible in that users are given a chance to access the resources they require at any time and pay only for computing. Cloud computing offers three types of cloud services [2] which are SaaS, PaaS and IaaS
1.2.1 Software as a service (SaaS)

SaaS is a cloud application that operates between different computers that are connected to the computers of users. This usually runs on an internet network. This service is of great advantage because it is not only able to scale usage needs in a dynamic manner but it also gives the users the ability to sign up and start using the services immediately. It also keeps data safe as it usually stores the data in the cloud and data may be accessed from any computer despite its distance so long as it is connected.

1.2.2 Platform as a service (PaaS)

PaaS provides users with a suitable environment to access all cloud resources that they need in the long run without the need to worry about the complexity and the cost of handling the software and hardware involved. This service is less complex as far as operating is concerned. Its applications are more developed and easily accessible and any new applications are directly uploaded to the cloud immediately.

1.2.3 Infrastructure as a service (IaaS)

IaaS is one of the most used services by cloud companies. This is because it provides them with all the cloud resources that they need in their operations. Some of the computing resources offered by this service include: data center [5], servers, networking and memory. The advantages of IaaS include: the services offered are innovative and flexible as they can be retrieved anytime they are needed; Users are provided with ready cloud hardware and hence dont need to incur extra charges to buy hardware: Lastly, the service can provide any amount of workload from the cloud hardware.
1.3 Cloud Computing Deployment Models

Cloud computing mostly is carried out in four server types from which users use to access the services and resources provided by the cloud [2].

1.3.1 Public Cloud

These are those types of clouds that are owned by companies which offer services over a public network. This means that everyone can access the resources of this cloud without necessarily having to purchase hardware or software. The administrator of the cloud may decide to offer the cloud service for free or use the license method like that of payment per user. In this case, the costs incurred are shared among all the users, so many users are likely to benefit more from public cloud compared to fewer users. Google is a good example of a public cloud.

1.3.2 Private Cloud

A type of cloud that can only be used by a particular organization. The only people who can access the services of this type of cloud authorized personnel. In the case of a private cloud, an organization assumes the role of the cloud provider and also the cloud consumer. This is done by using different departments of the same organization in that:

- One department acts as the provider of the cloud service whereby it is obligated with providing the cloud as per the requirement of the organization.

- Other departments are allowed to use the cloud service and hence act as the cloud consumers.
1.3.3 Community Cloud

This is a type of cloud that is shared by a community of organizations or firms. Only the users of the respective organizations can access the services of the cloud. Being a member of the community does not mean that one has direct access to the cloud and control of its resources. A community cloud simply means that the members of the community share the role of developing and maintaining the community cloud.

1.3.4 Hybrid Cloud

A hybrid cloud is a combination of two types of cloud services. Organizations that operate private clouds may have workloads which may require them to integrate public cloud into the system. Upon combining the public and the private cloud, there is formed a hybrid cloud. In this case, for instance, a cloud service consumer may choose to use the services of a private cloud for important data and use the services of a public cloud for less important data. Hybrid cloud usage can, however, be complex and difficult to create and maintain because private and public clouds have different operating environments and hence linking them together may be a challenge.

1.4 Prior Work

Different authors have given different techniques on load balancing [4] in cloud computing. Some of these techniques are stated below.

The Genetic Algorithm concepts have been discussed by various scientists. Kousik Dasgupta et al. Mandal, Dutta, Kumar, Dam [7] have on some occasions worked with the simple GA concepts. The interesting idea of the algorithm is based on the introduction of Processing Unit Vector and Job Unit Vector which was the main
activity behind the fitness function. Binary encoding is usually used by authors for representing chromosomes. Cloud Analyst is used as a stimulation tool and these authors responded to various outcomes with different numbers of data centers. These authors considered a similar priority for all the requests laid and came into a mutual conclusion that GA provided good QoS.

The method in which tasks are divided into sub tasks was proposed by Gupta and Singh [9]. The reason why these subdivisions were proposed was that of the parallel execution of activities on multiple machines. The GA operates in such a way that it distributes subtasks to different processors such that execution of a full task will be easier than working on different systems. The initial setup is made up of VMs (Virtual Machines) and requests respectively. Gupta and Singh never expanded much on the use of crossover although mutation is applied to fit chromosome in an attempt to find the best VM for the activity. The simulation of proposed system is done in CloudSim2.0 [12].

Based on the above survey [8], it can be concluded that without assigning the priority to the incoming requests, the system may lead to an unbalanced state that makes the system not suitable for real scenario. However, for less number of requests, it is not required to apply priority and even the traditional round robin algorithm [10] performs well. Only when the number of user requests increases, we need more sophisticated algorithms to balance the load evenly. Hence, this project seeks to come up with a fusion-based load aware algorithm which decides the suitable algorithm for tasks scheduling based on the number of user requests.

1.5 Problem Description

Cloud computing in general has various issues [6] listed below:
1. With the adoption of cloud computing techniques [3] by industries and corporate users the user count is increasing rapidly with increase in a number of cloud computing services. The companies have to increase data center count to serve the large number of incoming user requests to balance the heavy load on the servers.

2. Resource allocation among data centers i.e, allocation of virtual machines on data centers to provide enough resources keeping in mind the behavior and characteristics of data centers to provide high QoS at the resource level.

3. Current task allocation algorithms focus on static load balancing algorithm which balances the load when request load increases. However, this does not take into consideration of the previous behavior of the data center under high load. This will potentially leads us to design an efficient learning-based algorithms.

4. High loaded data centers have high failure probability to compute requests. With increase in load over data center request completion time increases which is not good for a user as well as a cloud provider. This leads to SLA (Service Level Agreement) [5] failure promised to the client.
CHAPTER 2

NARRATIVE

Cloud computing is a new trend in problem-solving and provides a reliable computing platform for intense computational tasks. Cloud is used for industries such as banking, trading and many e-commerce businesses to accommodate high request rate, high availability and prevents system failure. In case of failure, the requests are migrated to different reliable servers with letting the user knowing about it, providing fault tolerant behavior of the system. Other application zones include scientific research, weather forecasting report, satellite imaging and many more applications which require high computation capability, which is now possible without creating a private infrastructure.

Cloud computing is a reliable computing platform for computationally intensive or data intensive tasks. This has been accepted by many industrial giants of the software industry for their software solutions. Companies like Microsoft, Accenture, and Ericson have adopted cloud computing as their first choice for cheap and reliable computing. The tasks in the cloud need to be allocated in an efficient manner to provide high resource utilization and the least execution time for high performance.

Concepts of virtualization are used in the field of cloud which balances the load in the cloud. Virtualization refers to giving physical resources a logical name and it corresponds towards the physical resource each time the name is called out. Several users may need to use the cloud service at the same time and it is essential to serve all of them adequately within the given timeframe and the best service possible. For this reason, load balancing [13] that it balances the service given to each user on virtual machines evenly. Load balancing is an NP-Complete Problem [4] solving
method. This is attributed to more balancing occurring on the cloud infrastructure with an increase in the number of user requests.

2.1 Genetic Algorithm

Genetic Algorithm (GA), as shown in Fig. 2 below, has gained much popularity through the NP-Complete problem. GA belongs to a class of evolutionary techniques that solves problems based on the ability to detect natural evolution. The structure of basic GA as shown in Fig. 1 below, holds a number of simple concepts which are [1]:

- Population – This is a number of different possible solutions for a proposed problem.
- Chromosome – The individuals taking part in the population activities.
- Gene – This is a variable that is present in a chromosome.

Figure 1. Basic Genetic Algorithm Structure.

- Fitness Function – This is a type of an objective function that is used to check out the extent to which a solution is achieving the particular set aim.

Genetic Algorithm operates around four operators as follows:
Figure 2. Basic Genetic Algorithm Flow.

- Selection – Solutions which are the best fit for the problem at hand are selected.
- Crossover – More than one parent is selected for generation of a child.
- Gene – This is a variable that is present in a chromosome.
- Mutation – This refers to altering the gene value in a chromosome.
CHAPTER 3

SYSTEM DESIGN

3.1 Software and Hardware Requirements

- Operating Systems: Windows Vista SP2 or newer versions, Mac OS X Intel-based Mac running Mac OS X 10.8.3+, 10.9+, or linux systems
- Programming Language: Java version 8 or newer
- Framework: CloudSim Toolkit
- IDEs: NetBeans or Eclipse
- RAM: 128 MB
- Disk space: 124 MB for JRE; 2 MB for Java Update
- Processor: Minimum Pentium 2 266 MHz processor

3.2 Simulation Tool

There are plenty of simulation tools available when it comes to using a cloud simulator, but CloudSim outperforms the other frameworks with its user friendliness and robustness. It was built on top of Java programming language.

CloudSim 3.0 will be used for simulation in our project due to the following advantages with CloudSim.

- Experiments can be conducted to test the various parameters of the cloud system such as load balancing, performance improvement, and power consump-
tion, by extending the features of CloudSim without getting into the low-level implementation of the cloud infrastructure and services.

- Experiments can be repeated with minimum effort to improve the accuracy of the tests
- It has high degree of flexibility

After choosing the CloudSim, the features must be modified and extended to suit our project requirements. An effective scheduling algorithm needs to be developed to run the cloudlets.

3.3 CloudSim Layered Architecture

As shown in Fig. 3 below, this architecture provides services necessary in simulation and modeling of various cloud interfaces under different conditions for example bandwidth, memory, storage and VMs. It also acts as a host for the execution of applications and the monitoring of dynamic systems.

The user identity in the architecture provides various entities, for instance, including: the number of machines involved, the features of the machines, the applications running on the machines, VMs and the scheduling strategies.

3.3.1 The main components of the CloudSim framework

- Regions comes up with the locations in which the cloud dealers choose to provide the cloud services to their clients. There are six regions that represent six continents around the world in cloud analysis.

- Data centers design the environment of cloud services provided by various providers. It classifies hosts and servers that are either different or similar
in nature, based on the setting of the hardware. Data center characteristics provides necessary information on the settings of data center resources.

- Hosts create physical resources involved in a Cloudsim which may include storage and computing services.

- User base creates a group of users considered as a single unit in the simulation. Its main role is to create traffic for the simulation.

- Cloudlet comes up with the set of user requests. The name of the user, the Identity, the size of the request execution and the input as well as the output files are contained in the cloudblet. It creates the services of the cloud application. Cloudsim differentiates the complexity or simplicity of applications based on the requirements for computation. Each application has roles that it has to undertake for the entire life cycle.

- Service broker makes the decision on which data center should be used to meet requests from the users.

- VMM allocation policy comes up with the strategy to be used in the allocation of VMs to the dealer. It comes up with the time or spaces shared among users. It also comes up with a strategy on how to assign processor cores to VMs.

3.4 Proposed System Design

As shown in Fig. 4 below a fusion-based load aware algorithm assuming that we already calculated load threshold values for min load, intermediate load and max load respectively by taking current load into consideration.
3.5 Proposed System Implementation

The system will be implemented by using a fusion-based Load aware algorithm (FLA).

3.5.1 Methodology

We need to calculate the current load on the system in order to determine the threshold values for light load, intermediate load and heavy load respectively. If the current load on the cloud system is less than or equal to light load threshold value then we will be scheduling the user requests using traditional Round-robin scheduling algorithm. If the current load on the system is less than or equal to the intermediate load threshold value then we will be scheduling the user requests using basic genetic algorithm (BGA). Otherwise, we will schedule the tasks using priority-based genetic algorithm (PGA).

3.5.2 Round-robin Implementation

All the incoming user requests are divided among the VMs in this algorithm in equal portions and in circular order. The processing capabilities of each VM might differ from the other. Different user requests have different priorities and do not need to have the same job processing time. At some stage, some VMs may be heavily loaded and others remain idle. Time quantum plays a vital role in scheduling the user requests in Round-robin fashion. When time quantum is very large then RR Scheduling Algorithm is same as the FCFS Scheduling. and when the time quantum is too small then Round Robin Scheduling is known as Processor Sharing Algorithm.
3.5.3 Basic Genetic algorithm (BGA) Implementation

Genetic algorithm is a random searching method that has a better optimization ability and internal implicit parallelism. It can obtain and instruct the optimized searching space and adjust the searching direction automatically through the optimization method of probability.

After the first population is generated, it evolves better and better approximate solutions using the law of survival of the fittest from the generations. An individual is chosen in every generation based on the fitness of different individuals in certain problem domains. A new population representing a new solution set is produced when different individuals combine, cross, and vary by genetic operators in natural genetics. This project presents a scheduling strategy through the genetic algorithm based on the real situation of cloud computing.

3.5.3.1 Steps in the basic genetic algorithm (BGA)

Basic genetic algorithm is implemented as follows.

Step 1: Generate random population with n chromosomes

Step 2: Calculate the fitness value \( f(x) \) of every chromosome in the given population.

Step 3: Generate the new population by reiterating the following steps till the creation of new population is done.

Step 3.1: Select two parent individuals from the population according to the fitness value.

Step 3.2: By using the crossover probability, generate the new offspring by reforming the parents.

Step 3.3: With the probability of mutation, mutate the new child at some positions.

Step 3.4: accept the new offspring the part of next generation of population.
Step 4: replace the new generation as the current generation.
Step 5: Test if the stopping condition is satisfied then end the algorithm and return the individual with the highest fitness value.
Step 6: goto step 2.

3.5.4 Priority based Genetic Algorithm (PGA) Implementation

The system will be implemented by using Priority-based genetic algorithm (PGA). Figure ?? below describes the flow chart for the Priority-based Genetic Algorithm method.

Step 1: Calculate the priorities of incoming requests in the order of their time.
Step 2: Initialize the population based on the priority of jobs.
Step 3: Select the best chromosome based on fitness.
Step 4: Perform partially mapping crossover (PMX).
Step 5: Perform mutation by the use of swapping technique.
Step 6: Add the chromosome to a new population
Step 7: Check for the termination condition.
Step 8: End
Figure 3. Layered CloudSim architecture.
Figure 4. Fusion-based Load aware Resource Allocation on Cloud Infrastructure.
Figure 5. Flow Chart of Priority-based Genetic Algorithm Method.

- Initialize population based on priority of request i.e. sort cloudlets by length
- Select chromosomes based on fitness
- Two-Point Crossover
- Mutation (Swapping)
- New Population
- If the termination condition is met
  - True: End
  - False: Repeat process
CHAPTER 4

EVALUATION AND EXPECTED RESULTS

4.1 Software Requirements

- Operating Systems: Windows Vista SP2 or newer versions or Mac OS X Intel-based Mac running Mac OS X 10.8.3+, 10.9+ or Linux systems

- Programming Language: Java version 8 or newer

- Framework: CloudSim Toolkit

- IDEs: NetBeans or Eclipse

4.2 Experimental Setup

Several preliminary experiments have been conducted in order to assess the influence of Fusion-based Load aware algorithm (FLA) on cloud infrastructure using CloudSim 3.0. On several instances, proposed simulations have been carried out with the following varying numbers of Average. Peak Users:

- One Data Center, 25 VMs and Users in 100s

- Two Data Centers, 50 VMs and Users in 1000s

- One Data Center, 50 VMs and Users in 1000s

- Two Data Centers, 100 VMs and Users in 10000s
4.3 Expected Results

According to the above considerations, calculation of overall response time for each experimental setup has been done. The simulations will be conducted for Round-robin (RR), Basic genetic algorithm (BGA) and Priority-based genetic algorithm (PGA) and proposed Fusion load aware algorithm (FLA) for load balancing on cloud infrastructure using CloudSim 3.0. After calculation, the results will be presented in the form of graphs in the same order to make it easy for the users for interpretation. The results show that proposed algorithm works better for both underloaded and overloaded conditions.
BIBLIOGRAPHY AND REFERENCES


