Mobile Agent Community Pool – A New View on Load Balancing in Cloud Computing

Navaneetha.M#1, Chandriga.C#2, M.Shivakumar#3

#1,2Assistant Professor, CSE, Dr. TIT, VTU, India
#3Professor, CSE, Dr. TIT, VTU, India

ABSTRACT

With the advent of Cloud Computing, the era of technology is at its extreme. Accordingly several issues are under research. In this paper we have considered Load balancing a major challenging issue and proposed architecture with the application of mobile agents. Load balancing is a methodology to distribute workload across multiple computers, or other resources over the network links to achieve optimal resource utilization, maximize throughput, minimum response time, and avoid overload. The paper briefs out the load balancing algorithms and the mobile agents. The proposed architecture consists of a “Mobile Agent Community pool” which holds all the available agents. The agents in the pool are like Mobile Agents at user level, MA for type of cloud and service, for each load balancing algorithm a unique Mobile Agent is coded. In view of our proposed architecture an algorithm is defined making clear the placement of Mobile Agent community pool and the execution process that ensures the efficient load balancing and resource allocation strategy.

Keywords : Mobile Agents, Load Balancing, cloud computing.

I Introduction

Cloud computing is a distributed computing paradigm that focuses on providing a wide range of users with distributed access to scalable, virtualized hardware and/or software infrastructure over the internet [7]. Potentially it can make the new idea of ‘computing as a utility’ in the near future. Despite this technical definition cloud computing is in essence an economic model for a different way to acquire and manage IT resources. An organization needs to weigh cost, benefits and risks of cloud computing in determining whether to adopt it as an IT strategy. The availability of advance processors and communication technology has resulted the use of interconnected, multiple hosts instead of single high-speed processor which incurs cloud computing.

Recently, public cloud is made available as a pay per usage model while private cloud can be built with the infrastructure of the organization itself. Web Services, Google App Engine, and Microsoft Azure are examples of public cloud. The service provided by the public cloud is known as utility computing. As benefits, users can access this service “anytime, anywhere”, share data and collaborate more easily, and keep data safely in the infrastructure.

Although there are risks involved with releasing data onto third party servers without having the full control of it. At the state of the art Cloud Computing systems are logically subdivided into three different kinds:

IaaS Infrastructure as a Service, i.e. systems which offer a fully virtualized architecture based on a service oriented interface, as an example it is possible to create and deliver a (virtual) machine to a final user leaving him fully administrative control.

PaaS Platform as a Service, i.e. the system offers a virtualized platform, usually with a web interface, on which it is possible to develop applications, customize systems.

AaaS Application as a Service, i.e. the system hosts applications that can be accessed directly through a web interface (as an example Google Apps). In academic world Cloud computing is going to be integrated with the GRID context, which collects large computational resources in a single infrastructure. Solutions like [8], [9], [10] aim at transforming the actual HPC resources in Cloud Infrastructures dedicated to deliver virtualized infrastructures (Infrastructure as a Service).

Figure 1: Cloud computing service delivery models [12]

A. Deployment Models

Private Cloud: The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.
Community Cloud: The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise.

Public Cloud: The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

Hybrid Cloud: The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability.

The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability.

Load balancing is the process of distributing the load among various nodes and its request for resources. Here some methodology or technique need to be followed in balancing the equal distribution of load among various nodes and its resources in the cloud environment[1].

Load balancing is a technique that supports networks and resources with high throughput and minimum response time [2]. Dividing the traffic between servers, data can be sent and received without major delay. Different kinds of algorithms are available that helps traffic loaded between available servers.

Cloud vendors are based on automatic load balancing services, which allowed entities to increase the number of CPUs or memories for their resources to scale with the increased demands [3]. This service is optional and depends on the entity's business needs. Therefore load balancers served two important needs, primarily to promote availability of cloud resources and secondarily to promote performance.

In order to balance the requests of the resources it is important to recognize a few major goals of load balancing algorithms:

- a) Cost effectiveness
- b) Scalability and flexibility
- c) Priority

A. Load Balancing Reviews

Here a short literature review is carried on Existing load Balancing algorithms as stated by Jeno, Krishnamoorthy and Blessed prince[17].

Brief reviews of few existing load balancing algorithms are presented in the following:

I. Token Routing: The main objective of the algorithm [1][4][14], is to minimize the system cost by moving the tokens around the system. But in a scalable cloud system agents can not have the enough information of distributing the work load due to communication bottleneck. So the workload distribution among the agents is not fixed. The drawback of the token routing algorithm can be removed with the help of heuristic approach of token based load balancing. This algorithm provides the fast and efficient routing decision. In this algorithm agent does not need to have an idea of the complete knowledge of their global state and neighbor's working load. To make their decision where to pass the token they actually build their own knowledge base. This knowledge base is actually derived from the previously received tokens. So in this approach no communication overhead is generated.

II. Round Robin: In this algorithm [1][5][14], the processes are divided between all processors. Each process is assigned to the processor in a round robin order. The process allocation order is maintained locally independent of the allocations from remote processors. Though the work load distributions between processors are equal but the job processing time for different processes are not same. So at any point of time some nodes may be heavily loaded and others remain idle. This algorithm is mostly used in web servers where Http requests are of similar nature and distributed equally.

III. Randomized: Randomized algorithm is of type static in nature. In this algorithm [1][5] a process can be handled by a particular node n with a probability p. The process allocation order is maintained for each processor independent of allocation from remote processor. This algorithm works well in case of processes are of equal loaded. However, problem arises when loads are of different computational complexities. Randomized algorithm does not maintain deterministic
approach. It works well when Round Robin algorithm generates overhead for process queue.

IV. Central queuing: This algorithm [1][6] works on the principal of dynamic distribution. Each new activity arriving at the queue manager is inserted into the queue. When request for an activity is received by the queue manager it removes the first activity from the queue and ends it to the requester. If no ready activity is present in the queue the request is buffered, until a new activity is available. But in case new activity comes to the queue while there are unanswered requests in the queue the first such request is removed from the queue and new activity is assigned to it. When a processor load falls under the threshold then the local load manager sends a request for the new activity to the central load manager. The central manager then answers the request if ready activity is found otherwise queues the request until new activity arrives.

V. Connection mechanism: Load balancing algorithm [1] can also be based on least connection mechanism which is a part of dynamic scheduling algorithm. It needs to count the number of connections for each server dynamically to estimate the load. The load balancer records the connection number of each server. The number of connection increases when a new connection is dispatched to it, and decreases the number when connection finishes or timeout happens.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Nature</th>
<th>Environment</th>
<th>Process migration</th>
<th>Resource Utilization</th>
<th>Steadiness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token Routing</td>
<td>dynamic</td>
<td>decentralized</td>
<td>possible</td>
<td>more</td>
<td>unstable</td>
</tr>
<tr>
<td>Round Robin</td>
<td>static</td>
<td>decentralized</td>
<td>not possible</td>
<td>less</td>
<td>stable</td>
</tr>
<tr>
<td>Randomized</td>
<td>static</td>
<td>decentralized</td>
<td>not possible</td>
<td>less</td>
<td>stable</td>
</tr>
<tr>
<td>Central Queuing</td>
<td>dynamic</td>
<td>centralized</td>
<td>not possible</td>
<td>less</td>
<td>unstable</td>
</tr>
<tr>
<td>Least Connection</td>
<td>dynamic</td>
<td>centralized</td>
<td>not possible</td>
<td>less</td>
<td>stable</td>
</tr>
</tbody>
</table>

Table -1: Comparative Study of Load Balancing Algorithms

Thus randomness of load arrival from various end cause non uniform loads on the server. After studying various load balancing algorithms, it is necessary to execute at least two or three algorithms to observe where they are improving and lacking as well. In real life scenarios, executions of such algorithms are not possible in exact cloud computing environment. So to overcome our proposed Architecture uses the mobile agents for load sharing.

III Mobile Agents and its Advantages in Load Balancing

The term Mobile Agents is creating more curiosity, it can be viewed in terms of a normal Agent for example LIC Agent. The main task of these LIC Agents in an LIC Corporation is to find creat transparency in between the customer and the company, the LIC Agents are trained with all the terms and conditions of the corporation, and further Agents are intact with the customers, similarly here a software agent is mobile in nature, it holds the information of the intended source and it works irrespective of network topology.

They are two approaches that forms a new paradigm for distributed computing as stated by mohit mittal and Tarun Bhala [15] like RPC (Remote Procedure Call) and Mobile Agents. Here we are concerned with the Mobile Agents and its advantages in Load Balancing.

The use of Mobile agents is important based on the

- Consumption of bandwidth by mobile Agents
- And Continuous execution of these agents.

Mobile agents have the following unique properties[15]

- Adaptive Learning:
- Autonomy:
- Mobility:

Our proposed architecture uses mobile agents for load sharing, will be having some of the advantages as follows

A. Advantages of Mobile Agents Application in Cloud Computing for Load Balancing

As proposed by Parineeth in his general article – Mobile agents [16], here we wish to quote the advantages of the usage of these agents.

- Reduction in Network Load

Since the communication protocols are replaced by MA, these Agents perform the computation at the remote host and carry only the result. Hence it reduces the load on the network.

- Overcome Network Latency

No intermediate hosts are required for these agents, wherein the agents communicate directly.

- Protocol Encapsulation

The mobile agent code can encapsulate the protocol. When a protocol is upgraded, only the mobile agent has to be altered.

- Asynchronous and Autonomous Execution

Mobile agents operate asynchronously and executes autonomously without the intervention of the home machine. The home machine can reconnect at a later time and collect the agent.

- Fault Tolerance

Mobile agents react dynamically and autonomously to the changes in their environment [13], which makes them robust, and fault tolerant. They have the ability to distribute themselves in the network in such a way as to maintain the optimal configuration for solving the particular problem. If a host is being shutdown, all agents executing on that machine will be warned.
IV Usage of Mobile Agents

The proposed work is irrespective of type of cloud and/or services it provides. Here the cloud is assumed to be a black box from the end-user perspective. A Mobile Agent community pool is created, the hierarchical structure of the agent community is as in the figure, the pool consists of various categories of agents like,

a. Unique Mobile agents for each service of cloud
b. Separate Mobile agents for each type of cloud
c. Unique Mobile agents coded with load balancing algorithms.
   “Here each agent is coded with unique load balancing algorithm which is discussed in the previous section.
d. Mobile Agents in the network infrastructure (Agents for confidentiality and Integrity)
e. Mobile Agents at the end users (Agents for Authentication)

Figure 5: Hierarchical Structure of Mobile Agents.

V Proposed Architecture

In the proposed architecture, the cloud environment is considered to have three components namely client (end users), cloud service provider and Agent pool. The whole procedure can be summarized in following steps:

Step1: Client must authenticate with Mobile agent Community pool. This can be performed by username password or other mechanism.

Step2: After above authentication procedure, Master Agent in-charge of Agent community pool performs confidentiality and integrity check.

Step3: Mobile agents requests for resources through Master Agent. Master agent will intern dispatch all the mobile agents from the pool and send it for cloud service provider on the behalf of client according to requirement and service load.

Step4: When the Mobile Agents complete the task, they are returned back to the Mobile Agent community pool.

Figure 6: Mobile Agents Based Load Balancing in Cloud Computing.

In the above architecture, the end user for any type of service approaches the Agent pool, after a secure validation, the master agent of the pool fetches those free agents from the pool and assigns the service. Further the Mobile Agents perform the task as defined in the Algorithm.
A. Algorithm 1: User Registration.

Following steps are executed as a user initiates a service,

Step1: When user initiates the session, The User-Level Agents get initiated.

Step2: The User-Level Agents Communicate with Master Agent of The pool

Step3: The Authentication Procedure is followed,

If \( MA_{user} = \text{existing Reg-id} \)
Then
Registrant Access,
Else \( MA_{user} = \text{new-user} \)
Register

Step4: After authentication, the user-level agent gets the service by the Master Agent of the pool.

B. Algorithm 2: Mobile Agent Community Pool and Cloud

The sequence of steps that flows in-between the pool and cloud is as follows,

Step1: The User-Level Agent with the assigned Mobile Agents from the Community pool approaches the Service provider.

Step2: Confidentiality and integrity check are performed

a. An Mobile agent\( MA_{net} \) is fetched from the pool and assigned to perform the confidentiality check
b. Another Mobile agent\( MA_{net} \) is fetched from the pool and assigned to perform integrity check.

Step3: if \( MA_{user} = \text{Valid-Regid} \&\& MA_{Sec}=Valid&&MA_{Int}=Valid \)
Provide Key KEY,
Else
Claim false alarm, and return to pool.

Step4: if \( MA_{user}=\text{KEY} \&\& \text{KEY}=\text{Valid(User}\{\text{Regid,Sec,Int}\}) \)

a. Access the type of Cloud
b. Initiate the type of Service

Step5: Assign MA with suitable load balancing algorithm

a. If Access\( MA_{user}(\text{KEY}) \) = Dynamic
Then initiate the \( MA_{Dynamic} \)
\( MA_{Dynamic} = MA_{TR} \| MA_{CO} \| MA_{LC} \)
Else
Initiate the \( MA_{Static} \)
\( MA_{Static} = MA_{RR} \| MA_{Randomised} \)

Step6: if task completed, return the KEY.

Step7: Flush the task of all MA, and return to pool.

VI Conclusion

With the advent of cloud computing, many issues have been identified and is in research, accordingly we have considered load balancing as an serious issue, further in this paper we have proposed an architecture consisting of Mobile Agents. As a new paradigm the Mobile Agent community pool is implemented as a barrier between the end-user and the service provider.

The idea behind this Mobile Agent community pool consisting of pool of agents is to distribute the task and to reduce the load, further to create an image of the Cloud as a black box to the end user. The concept of security threats against the pool is essential.

REFERENCES


AUTHORS PROFILE

Navaneetha, Assistant Professor, Department of CSE, Dr. T. Thimmaiah Institute Of Technology, KGF. her area of interest is network security and had published many papers in national and international journals.

Chandriga.C, Assistant Professor, Department of CSE, Dr. T. Thimmaiah Institute Of Technology, KGF. her area of interest is Web Security and had published many papers in national and international journals.

Dr. M. Shiva Kumar, Professor and Head, Department of CSE, Dr. T. Thimmaiah Institute of technology, KGF. Obtained is doctoral degree in Network security and had presented papers in various conferences and published in various journals (National & international).