**Web Hosting On Cloud**

**ABSTRACT**

We address the problem of resource management for a large-scale cloud environment that hosts sites. Our contribution centers around outlining a distributed middleware architecture and presenting one of its key elements, a gossip protocol that meets our design goals: fairness of resource allocation with respect to hosted sites, efficient adaptation to load changes and scalability in terms of both the number of machines and sites. We formalize the resource allocation problem as that of dynamically maximizing the cloud utility under CPU and memory constraints. While we can show that an optimal solution without considering memory constraints is straightforward (but not useful), we provide an efficient heuristic solution for the complete problem instead. We evaluate the protocol through simulation and find its performance to be well aligned with our design goals.

**EXISTING SYSTEM:**

Resource Management is critical in cloud computing. With improper resource management, applications might experience network congestion, long time wait, CPU waist, overused CPU and memory, and security problems. To maximized cloud computing infrastructure utilization and minimize total cost of both the cloud computing infrastructure and running applications, resources need to be managed properly. To overcome this there are kinds of resources in the large-scale computing infrastructure need to be managed, CPU load, network bandwidth, disk quota, and even type of operating systems. To provide better quality of service, resources are provisioned to the users or applications, via load balancing mechanism, high availability mechanism and security and authority mechanism. To maximize cloud utilization, the capacity of application requirements shall be calculated so that minimal cloud computing infrastructure devices shall be procured and maintained. Given access to the cloud computing infrastructure, applications shall allocate proper resources to perform the computation with time cost and infrastructure cost minimized.

**PROPOSED SYSTEM:**

The proposed system considers the process of resource management for a large-scale cloud environment. Such an environment includes the physical infrastructure and associated control functionality that enables the provisioning and management of cloud services. The perspective we take is that of a cloud service provider, which hosts sites in a cloud environment. The cloud service provider owns and administrates the physical infrastructure, on which cloud services are provided. It offers hosting services to site owners through a middleware that executes on its infrastructure. Site owners provide services to their respective users via sites that are hosted by the cloud service provider.

This work contributes towards engineering a middleware layer that performs resource allocation in such a cloud environment, with the following design goals:

1) Performance objective: We consider computational and memory resources, and the objective is to achieve max-min fairness for computational resources under memory constraints.

2) Adaptability: The resource allocation process must dynamically and efficiently adapt to changes in the demand for cloud services.

3) Scalability: The Resource allocation process must be scalable both in the number of machines in the cloud and the number of sites that the cloud hosts. This means that the resources consumed per machine in order to achieve a given performance objective must increase sublinearly with both the number of machines and the number of sites.

**MODULE DESCRIPTION:**

# **Number of Modules**

After careful analysis the system has been identified to have the following modules:

1. **Resource Allocation For Cloud Environment (csp) Module.**
2. **Site Owner Module.**
3. **Site Manager Module.**
4. **Gossip Protocols Module.**

**1. Resource Allocation For Cloud Environment (csp)Module:**

The cloud service provider owns and administrates the physical infrastructure, on which cloud services are provided. The perspective we take is that of a cloud service provider, which hosts sites in a cloud environment. It offers hosting services to site owners through a middleware that executes on its infrastructure. For this work, we consider a cloud as having computational resources (i.e., CPU) and memory resources, which are available on the machines in the cloud infrastructure. The Resource allocation process will be scalable both in the number of machines in the cloud and the number of sites that the cloud hosts. This means that the resources consumed per machine in order to achieve a given performance objective must increase sublinearly with both the number of machines and the number of sites.

**2.Site Owner Module:**

Site owners provide services to their respective users via sites that are hosted by the cloud service provider. So the site owner will register their domain name for hosting their sites in cloud environment. Then the site owner will send request to site manager to develop entire site with particular domain name in order to host sites in cloud. Once the clouds service providers enable their domain name then the url wil be allocated in order to access the site by users.

**3.Site Manager Module:**

The architecture associates (one or more) site manager with each site(s). The site managers develop the site with particular domain name. Each site manager handles user requests to a particular site. It has two important components: a demand profiler and request forwarder. The demand profiler estimates the resource demand of each module of the site based on the request statistics.

**4.Gossip Protocol Module:**

We presented a gossip protocol that computes a heuristic solution to the resource allocation problem. This protocol qualitatively behaves as expected based on its design. For instance, regarding fairness, the protocol performs close to an ideal system for scenarios where the ratio of the total memory capacity to the total memory demand is large. More importantly, the simulations suggest that the protocol is scalable in the sense that all investigated metrics do not change when the system size (i.e., the number of machines) increases proportional to the external load (i.e., the number of sites).

**SOFTWARE REQUIREMENTS**:

 Operating System : Windows

 Technology : Java and J2EE

 Web Technologies : Html, JavaScript, CSS

 IDE : My Eclipse

 Web Server : Tomcat

 Tool kit : Android Phone

 Database : My SQL

 Java Version : J2SDK1.5

**HARDWARE REQUIREMENTS**:

 Hardware : Pentium

 Speed : 1.1 GHz

 RAM : 1GB

 Hard Disk : 20 GB

 Floppy Drive : 1.44 MB

 Key Board : Standard Windows Keyboard

 Mouse : Two or Three Button Mouse

 Monitor : SVGA