International Journal of Advanced Research in Engineering and Technology (IJARET)
Volume 10, Issue 2, March-April 2019, pp. 189-209. Article ID: IJARET_10_02_019
Available online at http://iaeme.com/Home/issue/IJARET?Volume=10&Issue=2
ISSN Print: 0976-6480 and ISSN Online: 0976-6499
© **IAEME** Publication

CLOUD COMPUTING AND LOAD BALANCING

Dr Syeda Gauhar Fatima
Professor ECE dept, Deccan College of Engineering and Technology,
Darussalam, Hyderabad, India

Syeda Kausar Fatima
Associate Professor, Shadan College of Engineering and Technology, Hyderabad, India

Dr Syed Abdul Sattar
Principal, Nawab Shah Alam Khan College of Engineering and Technology,
New Malakpet , Hyderabad, India

Naseer Ahmed Khan
Student, ECE Dept, Deccan College of Engineering and Technology,
Darussalam, Hyderabad, India

Syed Adil
Student, ECE Dept, Deccan College of Engineering and Technology,
Darussalam, Hyderabad, Inida

ABTRACT

Cloud Computing is utility software that has the ability to change the software company and making the software more attractive. Cloud Computing has altered the way IT world used to design software and hardware. The rise in web traffic and different services are increasing day by day making load balancing a big research topic. Cloud computing is technology which uses virtual machine as an alternative of physical machine to host, store and network the different components of computing. Load balancers are used for distributing loads to different virtual machines as an alternative of real machines in such a way that none of the machines gets burdened heavily or lightly. The load balancing needs to be done appropriately because disappointment in any one of the node can lead to inaccessibility of data.

Keywords: Cloud Computing, Static Load Balancer, Dynamic Load Balancer, Load Balancer Algorithms.

Cite this Article: Dr Syeda Gauhar Fatima, Syeda Kausar Fatima, Dr Syed Abdul Sattar, Naseer Ahmed Khan and Syed Adil, Cloud Computing and Load Balancing, *International Journal of Advanced Research in Engineering and Technology*, 10(2), 2019, pp. 189-209.
http://iaeme.com/Home/issue/IJARET?Volume=10&Issue=2

1. INTRODUCTION

Due to the extreme success of internet in last few years, computing resources is now more universally available and it enabled attainment of a new computing idea called Cloud Computing. Cloud Computing environment requires the traditional service providers to have two various ways. These are infrastructure and service providers. Infrastructure provider's arrangement of cloud platforms and lease resources according to usage. Service providers provide a resources from infrastructure providers to serve end users. Cloud Computing has tempt the giant companies, like Google, Amazon and Microsoft considered as a great effect in today's Information Technology companies. Cloud computing concept attracted in several feature .These are as follows:-

- Lower initial investment
- Easier to manage
- Scalability
- Deploy faster
- Location independent
- Device independent
- Reliability
- Security

In spite of the fact that cloud computing has shown Enough chances to the IT industry of today's world, but still there are number of trials that requires to be carefully addressed. Our aim is provide a best understanding of cloud computing.

2. CLOUD COMPUTING OVERVIEW

Cloud computing [1] is a technology for the internet and central remote servers to uphold applications and data. Cloud computing allows users to use resources without installation and access their personal files at any computer with internet access. This technology allows inefficient computing by centralizing storage, memory, processing and bandwidth.

Cloud computing is a model of network computing where a program or application runs on a connected one or more servers rather than on a local computing device such as a PC, tablet or smart phone. Like the Conservative client-server model a user connects with a server to execute a task. The difference with cloud computing is that the computing process may run on more than one connected computers can be, utilizing concept of virtualization. With virtualization is one or more physical servers can be configured and splits into multiple unattached "virtual servers, all functioning independently and seem to the user to be a single physical device. Such virtual servers is do not physically consist and can therefore be moved in all directions and flaky up or down on the fly without affecting the end user. The computing resources have become coarse, which give end user and operator avail including broad access across multiple devices, resource pooling, on-demand service, Fast elasticity and service surveying capability.

Cloud computing is a mechanism of distributed computing that focuses on discuss a wide range of users with distributed access to virtualized hardware and software infrastructure over the internet. It involves distributed computing virtualization, networking, web services and we software. Idea of cloud computing has focus interest of users towards of parallel, distributed and virtualization computing systems today. It appears as a popular solution to provide economical and easy access to expressed IT resources. Through virtualization, cloud computing is able to address with the same physical infrastructure a large client base with different computational needs. The rapid growth in the field of cloud computing also surges severe security apprehensions. Lack of security is the only hurdle in wide adoption of cloud computing.

2.1. Types of Cloud

2.1.1. Public Cloud

In Public cloud is obtainable for public use instead for a large industry and is owned by an association selling cloud services. Customer has no visibility and control Excess where the computing infrastructure is hosted. The computing infrastructure is joint among any organizations.

2.1.2. Private Cloud

The computing infrastructure is operated for the private use of an organization. The cloud probably managed by the organization or third party. Private clouds are more expensive and more protected when compared to public clouds. Private clouds may be either on or off premises. Externally hosted private clouds are also only used by one organization, but are hosted by third party focusing in cloud infrastructure. Superficially hosted private clouds are low-priced than On-premise private clouds.

2.1.3. Hybrid Cloud

Hybrid cloud syndicates multiple clouds (private community of public) where those clouds recollect their unique identities, but are destined together as a unit. A related term is Cloud Bursting. In Cloud bursting organization is use their own computing infrastructure for common usage, but access the cloud for high load requirements. This confirm that a sudden surge in computing necessity is handled elegantly. hybrid cloud may offer consistent or proprietary access to data and applications , as well as a application portability.

2.1.4. Community Cloud

Community cloud is one where the cloud has been systematized to serve a common function or resolution. For instance one organization or for several organization ,but they share conman concerns such as their mission, security, policies, regulatory compliance needs and so on.

3. CLASSIFICATION BASED UPON SERVICE PROVIDER

3.1 Infrastructure as a Service (IaaS)

Infrastructure as a service (IaaS) involve offering hardware linked facilities using the principles of cloud computing. IaaS provides a virtual-machine, virtual storage, disk image library, virtual infrastructure, raw block storage, and file or object storage, load balancer, IP addresses, firewalls, virtual local area networks and software providers stock these resources on -demand from their large pools installed in data centers bundles. The IaaS service provider manages all the infrastructure.

3.2 Platform as a Service (PaaS)

In the PaaS models, cloud providers deliver a computing platform, classically inclusive of programming language execution environment, operating system, database and web server. Application developers can change and run their software solutions on a cloud platform without the cost and intricacy of buying and handling the basic hardware and software layers. The service provider manages the cloud infrastructure, the operating systems and the enabling software.

3.3 Software as a Service (SaaS)

SaaS includes a complete software offering at the cloud. Users can access a software application hosted by the cloud venders on pay-per-use basis. In the business model using software as a service, users are provided access to application software and databases. SaaS is a complete operating environment with applications, management, and the user interface .Cloud providers handles the infrastructure and platforms that run the applications. SaaS is sometimes referred as "on demand software" and is commonly priced on a pay-per-use backbone. SaaS providers generally price applications using a subscription fee.

4. LOAD BALANCING IN CLOUD COMPUTING

Load balancing [2] is the technique of distributing the load between various resources in any system. Thus load require to be distributed over the resources in cloud-based architecture, so that each resources does almost the equal amount of work at any point of time. Basic requirement is to provide some techniques to balance requests to provide the solution of fast response for request. Cloud Load Balancers manage online traffic by distributing workloads between multiple servers and resources automatically. They maximize output, reduces response time, and avoid overload.

Some algorithms target at achieve higher output, some target at achieve lowest response time, some other target to achieve maximum resource utilization while some target at achieve a trade-off between all these metrics. Figure1 represent a framework underneath which various load balancing algorithms work in a cloud computing environment.

Load balancing is a general term used for allocating a larger processing load to smaller processing nodes for enhancing the overall performance of system. In a distributed system environment, it is the process of distributing load among various other nodes of distributed system to improve both resource utilization and job response time. A perfect load balancing algorithm should avoid overloading or under loading of any specific node. But, in case of a cloud computing environment the selection of load balancing algorithm is not easy because it includes additional restraints like security, reliability, throughput etc. So, the main goal of a load balancing algorithm in a cloud computing environment is to improve the response time of job by distributing the total load of system. The algorithm must also ensure that it is not overloading any specific node.

Load balancers can work in two ways: one is cooperative and non-cooperative. In cooperative, the nodes work simultaneously in order to achieve the common goal of optimizing the overall response time. In noncooperative mode, the tasks run self-sufficiently in order to improve the response time of local tasks.

Load balancing algorithms, in general, can be divided into two categories: static and dynamic load balancing algorithm. A static load balancing algorithm does not take into account the previous state or behavior of a node while distributing the load. On the other hand, a dynamic load balancing algorithm checks the previous state of a node while distributing the load. The dynamic load balancing algorithm is applied either as a distributed or non-distributed. The benefit of using dynamic load balancing is that if any node fails, it will not stop the system; it will only affect the system performance. In a dynamic load balanced system, the nodes can interact with each other creating more messages when compared to a non-distributed environment. However, selecting an appropriate server needs real time communication with the other nodes of the network and hence, generates a greater number of messages in the network. Dynamic load balancer uses policies for keeping track of updated information. Dynamic load balancers have four policies [3]:

Transfer policy-It is used when a selected job is needed for transfer from a local to a remote node.

Selection policy-It is used when processors exchange information between them. **Location Policy-**It is accountable for selecting a destination node for the transfer. **Information Policy-**It is used to maintain a record of all the nodes of the system.

The task of load balancing is shared among distributed nodes. In a distributed system, dynamic load balancing can be done in two different ways. In the distributed one, the dynamic load balancing algorithm is performed by all nodes present in the system and the task of scheduling is shared among all nodes. Whereas, in the undistributed one, nodes work independently in order to achieve a common goal. We will be using three criteria for comparing the load balancing algorithms:

Throughput - It can be estimated by calculating the total number of jobs executed within a fixed span of time without considering the virtual machine creation and destruction time.

Execution Time - It can be estimated by measuring the time taken for completing the following processes time for formation of virtual machines, processing time of a process and
In general, any cloud computing algorithm performs the following decision-making steps:

- Client requests the cloud coordinator.
- Cloud coordinator divides the task into cloudlet, and sends it to data centers.
- Data center coordinators work on scheduling the tasks. It also selects the algorithm, which will be used for scheduling tasks.

5. LOAD BALANCING BASICS

With this common terminology established, let's inspect the basic load balancing transaction. As depicted, the load balancer will typically sit in-line between the client and the hosts that provide the services the client wants to use. As with most things in load balancing, this is not a rule, but more of a best practice in a typical deployment. Let's also assume that the load balancer is already configured with a virtual server that points to a cluster containing of two service points. In this distribution situation, it is mutual for the hosts to have a return route that points back to the load balancer so that return traffic will be processed through it on its way back to the client.

The basic load balancing transaction is as follows:

1. The client attempts to attach with the service on the load balancer.
2. The load balancer receives the connection, and after determining which host should receive the connection, changes the destination IP (and possibly port) to match the service of the selected host (note that the source IP of the client is not touched).
3. The host accepts the connection and responds back to the original source, the client, via its default route, the load balancer.
4. The load balancer intercepts the return packet from the host and now changes the source IP (and possible port) to match the virtual server IP and port, and forwards the packet back to the client.
5. The client receives the return packet, be certain of that it came from the virtual server, and continues the process.

This very simple example is relatively upfront, but there are a couple of key elements to take note of. First, as far as the client knows, it sends packets to the virtual server and the virtual server responds—simple. Second, the NAT takes place. This is where the load balancer replaces the destination IP sent by the client (of the virtual server) with the destination IP of the host to which it has chosen to load balance the request. Step three is the second half of this process (the part that makes the NAT "bi-directional"). The source IP of the return packet from the host will be the IP of the host; if this address were not changed and the packet was simply forwarded to the client, the client would be receiving a packet from someone it didn't request one from, and would simply drop it. Instead, the load balancer, remembering the connection, rewrites the packet so that the source IP is that of the virtual server, thus solving this problem.

5.1 Basic Load Balancing Terminology

It would certainly help if everyone used the same lexicon; inappropriately, every vendor of load balancing devices (and, in turn, ADCs) seems to use different vocabulary. With a little clarification, however, the confusion adjacent this issue can easily be eased.

5.2. Node, Host, Member, and Server

Most load balancers have the concept of a node, host, member, or server; some have all four, but they mean different things. There are two basic concepts that they all try to express. One concept—usually called a node or server—is the idea of the physical server itself that will receive traffic from the load balancer. This is synonymous with the IP address of the physical server and, in the absence of a load balancer, would be the IP address that the server name (for example, www.example.com) would resolve to. For the remainder of this paper, we will discuss to this concept as the host.

The second concept is a member (sometimes, unfortunately, also called a node by some manufacturers). A member is usually a little more defined than a server/node in that it includes the TCP port of the actual application that will be receiving traffic. For instance, a server named www.example.com may resolve to an address of 172.16.1.10, which represents the server/node, and may have an application (a web server) running on TCP port 80, making the member address 172.16.1.10:80. Simply put, the member includes the definition of the application port as well as the IP address of the physical server. For the remainder of this paper, we will refer to this as the service.

Why all the complication? Because the distinction between a physical server and the application services running on it allows the load balancer to individually interact with the applications rather than the underlying hardware. A host (172.16.1.10) may have more than one service available (HTTP, FTP, DNS, and so on). By defining each application uniquely (172.16.1.10:80, 172.16.1.10:21, and 172.16.1.10:53), the load balancer can apply unique load balancing and health monitoring (discussed later) based on the services instead of the host. However, there are still times when being able to interact with the host (like low-level health monitoring or when taking a server offline for maintenance) is extremely convenient.

Remember, most load balancing-based technology uses some concept to represent the host, or physical server, and another to represent the services available on it— in this case, simply host and services.

5.3. Pool, Cluster, and Farm

Load balancing allows organizations to distribute inbound traffic across multiple back-end destinations. It is therefore a necessity to have the concept of a collection of back-end destinations. Clusters, as we will refer to them herein, although also known as pools or farms, are collections of similar services available on any number of hosts. For instance, all services that offer the company web page would be collected into a cluster called "company web page" and all services that offer ecommerce services would be collected into a cluster called "e-commerce."

The key element here is that all systems have a collective object that refers to "all similar services" and makes it easier to work with them as a single unit. This collective object—a cluster—is almost always made up of services, not hosts.

5.4. Virtual Server

Although not always the case, today there is little dissent about the term virtual server, or virtual. It is important to note that like the definition of services, virtual server usually includes the application port was well as the IP address. The term "virtual service" would be more in keeping with the IP:Port convention; but because most vendors use virtual server, this paper will continue using virtual server as well.

5.5. Putting It All Together

Putting all of these concepts together makes up the basic steps in load balancing. The load balancer presents virtual servers to the outside world. Each virtual server points to a cluster of services that exist in on one or more physical hosts.

6. ISSUE EFFECTING LOAD BALANCING IN CLOUD COMPUTING

There are many categories of load balancing techniques which are available for cloud computing.

These load balancing techniques are: geographical distribution, static and dynamic.

6.1. The geographical Distribution

The geographical distribution of the nodes matters a lot in the shared performance of any real time cloud computing systems, specifically in case of the big scaled applications like Twitter, Facebook etc. A well-distributed system of nodes in cloud environment is useful in handling fault tolerance and maintaining the efficiency of the system. Geographical load balancing (GLB) can be defined as a series of decisions about online assignment and/or migration of virtual machines (VMs) or computational tasks to geographically distributed datacenters in order to meet the service level agreements (SLAs) or service deadlines for VMs/tasks and to decrease the operational cost of the cloud system.

6.2. Static Load Balancing (SLB)

In static load balancing algorithms, the execution of the processors is determined at the beginning of the execution, it does not depend on current state of the system. The goal of static load balancing is to reduction the overall execution time of a synchronous program while lessening the communication delays. These algorithms are mostly suitable for homogeneous and stable environments and can produce better results .Some of the examples of static load balancing algorithms are: Randomized algorithm, Round Robin algorithm and Threshold algorithm.

6.3. Dynamic Load Balancing (DLB)

In dynamic load balancing algorithm, the decisions in load balancing are based on the current state of the system, no prior knowledge is needed. The major benefit of dynamic load balancing is that if someone node fails, it will not halt the system, it will only affect the performance of the system. These algorithms are more resilient than static algorithms, can easily adapt to alteration and provide better results in heterogeneous and dynamic environments. Dynamic load balancer uses morality for keeping the track of updated information.

Therein are four policies for dynamic load balancers: selection policy, transfer policy, location policy and information policy. The task of load balancing is shared among distributed nodes. In a distributed system, dynamic load balancing can be done in two different ways: distributed and non-distributed.

6.3.1. Distributed Dynamic Load Balancing (DDLB)

In the distributed one, the dynamic load balancing algorithm is implemented by all nodes present in the system and the task of scheduling is shared among them. The interaction among the nodes to achieve load balancing can take two forms: cooperative and non -cooperative.

6.3.2. Non-Distributed Load Balancing (NDLB)

In the non-distributed or undistributed, the nodes work personal in order to instate a common goal. Non distributed dynamic load balancing algorithms are ahead classified into two: centralized and semi-centralized.

6.3.2.1. Semi-distributed Dynamic Load Balancing (SDLB)

In semi-distributed dynamic load balancing, the nodes of the system are separations into clusters, where the load balancing in each cluster is of centralized form. A central node is elected in every cluster by appropriate election technique which takes care of load balancing within that cluster.

Therefore, the load balancing of all system is done via the central nodes of each cluster.

6.3.2.2. Centralized Dynamic Load Balancing (CDLB)

In centralized dynamic load balancing, the algorithm is only executed by a single node in the whole system i.e. central node. This node is perfectly responsible for load balancing of the whole system and rest of the nodes interacts only with the central node.

The residual part of this paper are organized as section 6 describes related work about various load balancing technique used in recently in soft computing, artificial antigens(AI) and other related subjects we conclude our paper and also provide direction for future improvement.

7. LITERATURE REVIEW IN LOAD BALANCING

Following load balancing techniques are currently prevalent in clouds:-

Decentralized Content Aware Load Balancing - H. Mehta et al. [4] proposed a new content aware load balancing policy named as workload and client aware policy (WCAP). It uses a unique and special property (USP) to stipulate the unique and special property of the requests as well as computing nodes. USP helps the scheduler to decide the best suitable node for the dispensation the requests. This strategy is implemented in a decentralized manner with low overhead. By using the content information to narrow down the search, this technique improves the searching performance and hence overall performance of the system. It also helps in reducing the idle time of the computing nodes hence improving their utilization.

Server-based Load Balancing for Internet Distributed Services - A. M. Nakai et al. [5] proposed a new server based load balancing policy for web servers which are distributed all over the world. It helps in reducing the service response times by using a protocol that limits the redirection of requests to the closest remote servers without overloading them. A middleware is described to implement this protocol. It also uses a heuristic to help web servers to endure overloads.

Join-Idle-Queue - Y. Lu et al. [6] proposed a Join Idle-Queue load balancing algorithm for dynamically scalable web services. This algorithm provides large scale load balancing with distributed dispatchers by, first load balancing idle processors across dispatchers for the availability of idle processors at each dispatcher and then, assigning jobs to processors to reduce average queue length at each processor. By removing the load balancing work from the critical path of request processing, it effectively reduces the system load, incurs no communication overhead at job arrivals and does not increase actual response time.

A Lock-Free Solution for Load Balancing in Multi-Core Environment - X. Liu et al. [7] proposed a lock-free multiprocessing load balancing solution that avoids the use of shared memory in contrast to other multiprocessing load balancing solutions which use shared memory and lock to maintain a user session. It is achieved by modifying Linux kernel. This solution helps in improving the overall performance of load balancer in a multi-core environment by running multiple load-balancing processes in one load balancer.

Scheduling Strategy on Load Balancing of Virtual Machine Resources - J. Hu et al. [8] proposed a scheduling strategy on load balancing of VM resources that uses historical data and current state of the system. This strategy achieves the best load balancing and reduced dynamic migration by using a genetic algorithm. It helps in resolving the issue of load-imbalance and high cost of migration thus achieving better resource utilization.

Performance Evaluation of Web Servers Using Central Load Balancing Policy Over Virtual Machines on Cloud - A. Bhadani et al. [9] proposed a Central Load Balancing Policy for Virtual Machines (CLBVM) that balances the load evenly in a distributed virtual machine/cloud computing environment. This policy improves the overall performance of the system but does not consider the systems that are fault-tolerant.

Load Balancing Strategy for Virtual Storage - H. Liu et al. [10] proposed a load balancing virtual storage strategy (LBVS) that provides a large scale net data storage model and Storage as a Service model based on Cloud Storage. Storage virtualization is achieved using an architecture that is three-layered and load balancing is achieved using two load balancing modules. It helps in improving the efficiency of concurrent access by using replica balancing further reducing the response time and enhancing the capacity of disaster recovery. This strategy also helps in improving the use rate of storage resource, flexibility and robustness of the system.

A Task Scheduling Algorithm Based on Load Balancing - Y. Fang et al. [11] discussed a twolevel task scheduling mechanism based on load balancing to meet dynamic requirements of users and obtain a high resource utilization. It achieves load balancing by first mapping tasks to virtual machines and then virtual machines to host resources thereby improving the task response time, resource utilization and overall performance of the cloud computing environment.

Honeybee Foraging Behavior - Richteret al. [12] investigated a decentralized honeybee-based load balancing technique that is a nature-inspired algorithm for self-organization. It achieves global load balancing through local server actions. Performance of the system is enhanced with increased system diversity but throughput is not increased with an increase in system size. It is best suited for the conditions where the diverse population of service types is required.

A Dynamic Biased Random Sampling Scheme for Scalable and Reliable Grid Networks Rahmeh et al. [13] investigated a distributed and scalable load balancing approach that uses random sampling of the system domain to achieve self-organization thus balancing the load across all nodes of the system. The performance of the system is improved with high and similar population of resources thus resulting in an increased throughput by effectively utilizing the increased system resources. It is degraded with an increase in population diversity.

Active Clustering - M. Randles et al. [14] investigated a self-aggregation load balancing technique that is a self-aggregation algorithm to optimize job assignments by connecting similar services using local re-wiring. The performance of the system is enhanced with high resources thereby increasing the throughput by using these resources effectively. It is degraded with an increase in system diversity.

Load Balancing Mechanism Based on Ant Colony and Complex Network Theory - Z. Zhang et al. [15] proposed a load balancing mechanism based on ant colony and complex network theory in an open cloud computing federation. It uses small-world and scale-free characteristics of a complex network to achieve better load balancing. This technique overcomes heterogeneity, is adaptive to dynamic environments, is excellent in fault tolerance and has good scalability hence helps in improving the performance of the system.

Two-Phase Load Balancing Algorithm (OLB + LBMM) - S.-C. Wang et al. [16] proposed a two- phase scheduling algorithm that combines OLB (Opportunistic Load Balancing) and LBMM (Load Balance Min-Min) scheduling algorithms to utilize better executing efficiency and maintain the load balancing of the system. OLB scheduling algorithm, keeps every node in working state to achieve

the goal of load balance and LBMM scheduling algorithm is utilized to minimize the execution time of each task on the node thereby minimizing the overall completion time. This combined approach hence helps in an efficient utilization of resources and enhances the work efficiency.

Event-driven - V. Nae et al. [17] presented an eventdriven load balancing algorithm for real-time Massively Multiplayer Online Games (MMOG). This algorithm after receiving capacity events as input, analyzes its components in context of the resources and the global state of the game session, thereby generating the game session load balancing actions. It is capable of scaling up and down a game session on multiple resources according to the variable user load but has occasional QoS breaches.

Carton - R. Stanojevic et al. [18] proposed a mechanism for cloud control named as CARTON that unifies the use of LB and DRL. Load Balancing is used to equally distribute the jobs to different servers so that the associated costs can be minimized and Distributed Rate Limiting is used to make sure that the resources are distributed in a way to keep a fair resource allocation. DRL also adapts to server capacities for the dynamic workloads so that performance levels at all servers are equal. With very low computation and communication overhead, this algorithm is simple and easy to implement.

Compare and Balance - Y. Zhao et al. [19] addressed the problem of intra-cloud load balancing amongst physical hosts by adaptive live migration of virtual machines. A load balancing model is designed and implemented to reduce virtual machines' migration time by shared storage, to balance load amongst servers according to their processor or IO usage, etc. and to keep virtual machines' zero-downtime in the process. A distributed load balancing algorithm COMPARE AND BALANCE is also proposed that is based on sampling and reaches an equilibrium very fast. This algorithm assures that the migration of VMs is always from high-cost physical hosts to lowcost host but assumes that each physical host has enough memory which is a weak assumption.

Vector Dot - A. Singh et al. [20] proposed a novel load balancing algorithm called VectorDot. It handles the hierarchical complexity of the data-center and multidimensionality of resource loads across servers, network switches, and storage in an agile data center that has integrated server and storage virtualization technologies. VectorDot uses dot product to distinguish nodes based on the item requirements and helps in removing overloads on servers, switches and storage nodes.

8. METRICS FOR LOAD BALANCING IN CLOUDS

The existing load balancing techniques in clouds, consider various parameters like performance, response time, scalability, throughput, resource utilization, fault tolerance, migration time and associated overhead. But, for an energy-efficient load balancing, metrics like energy consumption and carbon emission should also be considered [21].

Performance - Performance is used to check the efficiency of the system. It has to be improved at a reasonable cost e.g. reduce response time while keeping acceptable delays.

Resource Utilization - Resource Utilization is used to check the utilization of resources. It should be optimized for an efficient load balancing.

Scalability - Scalability is the ability of an algorithm to perform load balancing for a system with any finite number of nodes. This metric should be improved.

Response Time - Response Time is the amount of time taken to respond by a particular load balancing algorithm in a distributed system. This parameter should be minimized.

Fault Tolerance - Fault Tolerance is the ability of an algorithm to perform uniform load balancing in spite of arbitrary node or link failure. The load balancing should be a good fault-tolerant technique.

Migration time - Migration time is the time to migrate the jobs or resources from one node to other. It should be minimized in order to enhance the performance of the system.

Energy Consumption (EC) - Energy Consumption determines the energy consumption of all the resources in the system. Load balancing helps in avoiding overheating by balancing the workload across all the nodes of a Cloud, hence reducing energy consumption.

Carbon Emission (CE) - Energy Consumption calculates the carbon emission of all the resources in the system. As energy consumption and carbon emission go hand in hand, the more the energy consumed, higher is the carbon footprint. So, for an energy-efficient load balancing solution, it should be reduced.

9. GOALS OF LOAD BALANCING

The goals of load balancing are [22]:

- To improve the performance considerably
- To have a backup plan in case the system fails even partly
- To maintain the system firmness
- To accommodate future alteration in the system

10. FEATURES OF LOAD BALANCING

Hardware and software load balancers may have a variety of special features. The fundamental feature of a load balancer is to be able to distribute incoming requests over a number of backend servers in the cluster according to a scheduling algorithm. Most of the following features are vendor specific [23]:

Asymmetric Load: A ratio can be manually assigned to cause some backend servers to get a greater share of the workload than others. This is sometimes used as a crude way to account for some servers having more capacity than others and may not always work as desired.

Priority Activation: When the number of available servers drop below a certain number, or load gets too high, standby servers can be brought online.

SSL Offload and Acceleration: Depending on the workload, processing the encryption and authentication requirements of an SSL request can become a major part of the demand on the Web Server's CPU; as the demand increases, users will see slower response times, as the SSL overhead is distributed among Web servers. To remove this demand on Web servers, a balancer can terminate SSL connections, passing HTTPS requests as HTTP requests to the Web servers.

Distributed Denial of Service (DDoS) Attack Protection: load balancers can provide features such as SYN cookies and delayed-binding (the back-end servers don't see the client until it finishes its TCP handshake) to mitigate SYN flood attacks and generally offload work from the servers to a more efficient platform.

HTTP Compression: reduces amount of data to be transferred for HTTP objects by utilizing zip compression available in all modern web browsers. The larger the response and the further away the client is, the more this feature can improve response times.

The compromise is that this feature puts additional CPU demand on the Load Balancer and could be done by Web servers as an alternative.

TCP Offload: different vendors use different terms for this, but the idea is that normally each HTTP request from each client is a different TCP connection. This feature utilizes HTTP/1.1 to consolidate multiple HTTP requests from multiple clients into a single TCP socket to the back-end servers.

TCP Buffering: the load balancer can buffer responses from the server and spoon-feed the data out to slow clients, allowing the web server to free a thread for other tasks faster than it would if it had to send the entire request to the client directly.

Direct Server Return: an option for asymmetrical load distribution, where request and reply have different network paths.

Health Checking: the balancer polls servers for application layer health and removes failed servers from the pool.

HTTP Caching: the balancer stores static content so that some requests can be handled without contacting the servers.

Content Filtering: some balancers can arbitrarily modify traffic on the way through.

HTTP Security: some balancers can hide HTTP error pages, remove server identification headers from HTTP responses, and encrypt cookies so that end users cannot manipulate them.

Priority Queuing: also known as rate shaping, the ability to give different priority to different traffic.

Content-aware Switching: most load balancers can send requests to different servers based on the URL being requested, assuming the request is not encrypted (HTTP) or if it is encrypted (via HTTPS) that the HTTPS request is terminated (decrypted) at the load balancer.

Client Authentication: authenticate users against a variety of authentication sources before allowing them access to a website.

Programmatic Traffic Manipulation: at least one balancer allows the use of a scripting language to allow custom balancing methods, arbitrary traffic manipulations, and more.

Firewall: direct connections to backend servers are prevented, for network security reasons Firewall is a set of rules that decide whether the traffic may passthrough an interface or not.

Intrusion Prevention System: offer application layer security in addition to network/transport layer offered by firewall security.

11. TYPES OF LOAD BALANCING ALGORITHMS

Depending on who started the process, load balancing algorithms can be of three categories [24]:

Sender Initiated: If the load balancing algorithm is initialized by the sender

Receiver Initiated: If the load balancing algorithm is initiated by the receiver

Symmetric: It is the combination of both sender initiated and receiver initiated

Depending on the current state of the system, load balancing algorithms can be divided into 2 categories:

Static: It doesn't depend on the current state of the system. Previous knowledge of the system is needed.

Dynamic: Decisions on load balancing are based on current state of the system. No previous knowledge is needed. So it is better than static approach.

Here we will discuss on various dynamic load balancing algorithms for the clouds of different sizes.

11.1. STATIC LOAD BALANCERS

11.1.1. Round-Robin Load Balancer

It is a fixed load balancing algorithm [25], which does not take into account the previous load state of a node at the time of assigning jobs. It uses the round robin scheduling algorithm for assigning jobs. It chooses the first node randomly and then, assigns jobs to all other nodes in a round robin manner. This algorithm will not be suitable for cloud computing because some nodes might be deeply loaded and some are not. Since the running time of any process is not known prior to execution, there is a possibility that nodes may get profoundly loaded.

Hence, weighted round-robin algorithm was proposed to solve this problem. In this algorithm, each node is assigned a specific weight. Depending on the weight assigned to the node, it would receive suitable number of requests.

If the weight assigned to all the nodes are equal, then each node will receive same traffic. In cloud computing system, precise prediction of execution time is not possible therefore, this algorithm is not preferred.

11.1.2. Min-Min

It is a static load balancing algorithm [26]. So, all the information related to the job is available in advance. Some terminology related to static load balancing:

ETC: The expected running time of the job on all nodes are stored in an ETC (expected time of compute) matrix [27].

If a job is not executable on a particular node, the entry in the ETC matrix is set to infinity.

OLB: It is an opportunistic Load Balancing, in which each job is assigned to the node in an indiscriminate order, irrespective of the ETC on the nodes. It provides load balance schedule but it results in very poor make-span [27].

MET: It is a Minimum Execution Time algorithm. In this, each job is allocated to the node which has the lowest execution time as mentioned in ETC table, regardless of the current load on that processor. MET tries to find the greatest job-processor pair, but it does not take into consideration the current load on the node. This algorithm increases the make-span to some extent, but it causes a severe load imbalance [27].

MCT: It is a Lowest Completion Time algorithm which allocates jobs to the node based on their smallest completion time. The completion time is calculated by adding the expected execution time of a job on that node with node's ready time. The node with the lowest completion time for that particular job is selected. But this algorithm considers the job only one at a time [27].

Min-Min algorithm[28, 29] begins with a set of all unallocated jobs. First of all, minimum completion time for all jobs is calculated. The job with smallest completion time is selected. Then, the node which has the smallest completion time for all jobs is selected. Finally, the selected node and the selected job are charted. The ready time of the node is updated. This process is repeated until all the unassigned jobs are assigned. The benefit of this algorithm is that the job with the lowest execution time is executed. The disadvantage of this algorithm is that some jobs may experience starvation.

Max-Min: Max-Min[28, 29] is almost similar as the min-min algorithm except the following: after finding out minimum completion time of jobs, the maximum value is selected. The machine that has the minimum completion time for all the jobs is selected. Finally the selected node and the selected job are charted. Then the ready time of the node is updated by adding the execution time of the assigned task.

Load Balance Min-Min: LBMM [16] is a fixed load balancing algorithm. This algorithm implements load balancing among nodes by considering it as a scheduling problem. The main aim of this algorithm is to reduce the make -span, which is calculated as the maximum of the completion times of all the jobs scheduled on their respective resources. This algorithm performs the following steps for scheduling the jobs on the nodes.

- Execute the Min-Min scheduling algorithm and calculate the make-span.
- Select the node with the highest make-span value.
- Corresponding to that node, select the job with minimum execution time.
- The completion time of the selected job is calculated for all the resources.
- Maximum completion time of the selected job is compared with the make-span value. If it is less, the selected job is allocated to the node, which has the maximum completion time. Else, the next maximum completion time of the job is selected and the steps are repeated.
- The process stops if all the nodes and all the jobs are assigned.
- In scenarios where the number of small tasks is more than the number of large tasks in meta-task, this algorithm achieves better performance. This algorithm does not consider low and high machine heterogeneity and task heterogeneity.

11.1.3. Load Balance Max-Min-Max

S.-C. Wang et al. [16] proposed a two-phase scheduling algorithm that combinesOpportunistic Load Balancing (OLB) and Load Balance Min-Min (LBMM) scheduling algorithms. OLB scheduling algorithm keeps every node in working state to achieve the goal of load balancing and LBMM scheduling algorithm is utilized to minimize the execution time of each task on the node, thereby minimizing the overall completion time. This combined approach helps in efficient utilization of resources. This algorithm performs the following steps for load balancing:

- Average completion time of each task for all the nodes is calculated.
- To select the task with maximum average completion time.
- To select an unallocated node with lowest completion time that should be less than the maximum average completion time for the selected task. Then, the tasks dispatched to the selected node for computation.
- If all the nodes are already assigned, re-evaluate by considering both assigned and unassigned nodes. Minimum completion time is computed as:
- Minimum completion time of the assigned node is the sum of the minimum completion time for all the tasks assigned to that node and minimum completion time of the current task.
- Minimum completion time of the assigned node is the minimum completion time of the current task.

- Repeat step 2 to step 4, until all tasks are executed. This gives better results than the above discussed algorithms.

11.2. DYNAMIC LOAD BALANCERS EQUALLY SPREAD CURRENT EXECUTION

Similarly Spread current execution [30] is a dynamic load balancing algorithm, which handles the process with significance. It determines the priority by checking the size of the process. This algorithm allocates the load randomly by first checking the size of the process and then transferring the load to a Virtual Machine, which is lightly loaded. The load balancer spreads the load on to different nodes, and hence, it is known as spread spectrum technique.

11.2.1. Throttled Load Balancer

Throttled load balancer [30] is a dynamic load balancing algorithm. In this algorithm, the client first requests the load balancer to find a suitable Virtual machine to perform the required operation.

In Cloud computing, there may be multiple instances of virtual machine. These virtual machines can be grouped based on the type of requests they can handle. Whenever a client sends a request, the load balancer will first look for that group, which can handle this request and assign the process to the casually loaded instance of that group.

11.2.2. Honeybee Foraging Algorithm

The central idea behind the Honeybee Foraging algorithm[14] is derived from the behavior of honeybees. There are two kinds of honeybees: finders and reapers. The finder honeybees first goes open-air of the honey comb and find the honey sources. After finding the source, they back to the honey comb and do a waggle dance indicating the quality and quantity of honey obtainable. Then, reapers go outside and reap the honey from those sources. After collecting, they back to beehive and does a waggle dance. This dance indicates how much food is left. M. Randles proposed a decentralized honeybee based algorithm for self-organization. In this case, the servers are grouped as virtual server and each virtual server have a process queue. Each server, after processing a request from its queue, calculates the profit which is analogous to the quality that the bees show in their waggle dance. If profit is high, the server stays else, it returns to the forage. This algorithm requires that each node to maintain a separate queue. This calculation of profit on each node causes additional overhead.

The drawback of this algorithm is that, it does not show any significant improvement in throughput, which is due to the additional queue and the computation overhead.

11.2.3. Biased Random Sampling

Biased Random Sampling [14] is a dynamic load balancing algorithm. It uses random sampling of system domain to achieve self-organization thus, balancing the load across all nodes of system.

In this algorithm, a virtual graph is constructed with the connectivity of each node representing the load on server. Each node is represented as a vertex in a directed graph and each in-degree represents free resources of that node.

Whenever a client sends a request to the load balancer, the load balancer assigns the job to the node which has at least one in-degree. Once a job is allocated to the node, the in-degree of that node is decremented by one. After the job is completed, the node creates an incoming edge and increments the in-degree by one. The addition and deletion of processes is done by the process of random sampling.

Each process is characterized by a parameter know as threshold value, which indicates the maximum walk length.

A walk is defined as the traversal from one node to another until the destination is found. At each step on the walk, the neighbor node of current node is selected as the next node.

In this algorithm, upon receiving the request by the load balancer, it would select a node randomly and compares the current walk length with the threshold value. If the current walk length is equal to or greater than the threshold value, the job is executed at that node. Else, the walk length of the job is incremented and another neighbor node is selected randomly. The act is degraded as the number of servers increase due to additional overhead for computing the walk length.

11.2.4. Active Clustering

Active Clustering [14] is a clustering based algorithm which introduces the concept of clustering in cloud computing. In cloud computing there are many load balancing algorithms available. Each algorithm has its own compensations and drawbacks. Depending on the requirement, one of the algorithms is used. The performance of an algorithm can be enhanced by making a cluster of nodes. Each cluster can be assumed as a group. The principle behind active clustering is to group similar nodes together and then work on these groups.

The process of creating a cluster orbits around the concept of match maker node. In this process, first no deselects a neighbor node called the matchmaker node which is of a different type. This matchmaker node makes connection with its neighbor which is of same type as the initial node. Finally the matchmaker node gets detached. This process is followed iteratively. The performance of the system is enhanced with high obtainability of resources, thereby increasing the output. This increase in output is due to the efficient utilization of resources.

11.2.5. Join-Idle Queue

Join-Idle Queue[6] is basically used for large-scale systems. It uses distributed dispatchers by first load balancing the idle processors across dispatchers and then assigning jobs to processors to reduce average queue length at each processor. The disadvantage of this algorithm is that it is not scalable. Y. Lua et al[6] proposed this load balancing algorithm for dynamically scalable web services. It effectively reduces the system load, incurs no communication overhead at job arrivals and does not increase actual response time. It can perform close to optimal when used for web services. However, it cannot be used for today's dynamic-content web services due to the scalability and reliability.

12. LOAD BALANCING CHALLENGES IN THE CLOUD COMPUTING

Although cloud computing has been broadly accepted. Research in cloud computing is still in its early stages, and some scientific tests remain unresolved by the scientific community, mainly load balancing challenges [31].

Automated Service Provisioning: A key feature of cloud computing is elasticity, resources can be allocated or released automatically. How then can we use or release the resources of the cloud, by keeping the same performance as traditional systems and using optimal resources?

Virtual Machines Migration: With virtualization, an entire machine can be seen as a file or set of files, to unload a physical machine heavily loaded, it is possible to move a virtual machine between physical machines. The main objective is to distribute the load in a datacenter or set of datacenters. How then can we dynamically distribute the load when moving the virtual machine to avoid bottlenecks in Cloud computing systems?

Energy Management: The aids that advocate the adoption of the cloud is the economy of scale. Energy saving is a key point that allows a global economy where a set of global resources will be supported by reduced providers rather that each one has its own resources. How then can we use a part of datacenter while keeping acceptable performance?

Stored Data Management: In the last decade data stored across the network has an exponential increase even for companies by outsourcing their data storage or for individuals, the management of data storage or for individuals, the management of data storage becomes a major challenge for cloud computing. How can we distribute the data to the cloud for optimum storage of data while maintaining fast access?

Emergence of Small Data Centers for Cloud Computing: Small datacenters can be more beneficial, cheaper and less energy consumer than large datacenter. Small providers can deliver cloud computing services leading to geo -diversity computing. Load balancing will become a problem on a global scale to ensure an adequate response time with an optimal distribution of resources.

13. CONCLUSIONS AND FUTURE WORK

This paper is based on cloud computing technology which has a very vast potential and is still unexplored. The abilities of cloud computing are limitless. Cloud computing provides everything to the user as a service which includes platform as a service, application as a service, infrastructure as a service.

One of the major issues of cloud computing is load balancing because overloading of a system may lead to poor performance which can make the technology ineffective. So there is always a requirement of efficient load balancing algorithm for efficient utilization of resources. Our paper emphases on the various load balancing algorithms and their applicability in cloud computing environment.

We first considered the algorithms as static and dynamic. Then we analyzed the various algorithms which can be applied in static environments. After that we described the various dynamic load balancing algorithms. For solving any particular problem some special conditions need to be applied. So we have discussed some additional algorithms which can help in solving some sub-problems in load balancing which are applicable to cloud computing. In our future work we will analyze the algorithms with numerical analysis and simulation.

14. ACKNOWLEDGMENT

We would like to express our gratitude to Dr. Kalyani Mali, Head of Department, Computer Science and Engineering of University of Kalyani. Without her assistance and guidance, we would not have been able to make use of the university's infrastructure and laboratory facilities for conducting our research.

REFERENCES

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee et al. "A View of Cloud Computing." *Communications of the ACM* 53, No. 4 (2010): 50-58.

[2] Ananth Rao, Karthik Lakshminarayanan, Sonesh Surana, Richard Karp, and Ion Stoica. "Load Balancing in Structured P2P Systems." In *Peer-to-Peer Systems II*, pp. 68-79. Springer Berlin Heidelberg, 2003.

[3] N. Ajith Singh, M. Hemalatha, ―An Approach on Semi Distributed Load Balancing Algorithm for Cloud Computing Systems‖ International Journal of Computer Applications Vol-56 No.12 2012.

[4] Mehta, H., Priyesh Kanungo, and Manohar Chandwani. "Decentralized Content Aware Load Balancing Algorithm for Distributed Computing Environments." In *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, pp. 370-375. ACM, 2011.

[5] Nakai, Alan Massaru, Edmundo Madeira, and Luiz E. Buzato. "Load Balancing for Internet Distributed Services Using Limited Redirection Rates." In *Dependable Computing (LADC), 2011 5th Latin-American Symposium on*, pp. 156-165. IEEE, 2011.

[6] Lu, Yi, QiaominXie, Gabriel Kliot, Alan Geller, James R. Larus, and Albert Greenberg. "JoinIdle-Queue: A Novel Load Balancing Algorithm for Dynamically Scalable Web Services." *Performance Evaluation* 68, no. 11 (2011): 1056-1071.

[7] Liu, Xi, Lei Pan, Chong-Jun Wang, and Jun-Yuan Xie. "A Lock-Free Solution for Load Balancing in Multi-Core Environment." In *Intelligent Systems and Applications (ISA), 2011 3rd International Workshop on*, pp. 1-4. IEEE, 2011.

[8] Hu, Jinhua, Jianhua Gu, Guofei Sun, and Tianhai Zhao. "A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment." In *Parallel Architectures, Algorithms and Programming (PAAP), 2010 Third International Symposium on*, pp. 89-96. IEEE, 2010.

[9] Abhay Bhadani, and Sanjay Chaudhary. "Performance Evaluation of Web Servers Using Central Load Balancing Policy Over Virtual Machines on Cloud." In *Proceedings of the Third Annual ACM Bangalore Conference*, p. 16. ACM, 2010.

[10] Liu, Hao, Shijun Liu, Xiangxu Meng, Chengwei Yang, and Yong Zhang. "Lbvs: A load Balancing Strategy for Virtual Storage." In *Service Sciences (ICSS), 2010 International Conference on*, pp. 257-262. IEEE, 2010.

[11] Fang, Yiqiu, Fei Wang, and JunweiGe. "A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing." In *Web Information Systems and Mining*, pp. 271-277. Springer Berlin Heidelberg, 2010.

[12] Richter, M. Raveret. "Social Wasp (Hymenoptera: Vespidae) Foraging Behavior." *Annual Review of Entomology* 45, No. 1 (2000): 121-150.

[13] Rahmeh, O. Abu, P. Johnson, and A. Taleb-Bendiab. "A Dynamic Biased Random Sampling Scheme for Scalable and Reliable Grid Networks." *INFOCOMP Journal of Computer Science* 7, No. 4 (2008): 1-10.

[14] Martin Randles, David Lamb, and A. Taleb-Bendiab. "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing." In *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on*, pp. 551-556. IEEE, 2010.

[15] Zhang, Zehua, and Xuejie Zhang. "A Load Balancing Mechanism Based on Ant Colony and Complex Network Theory in Open Cloud Computing Federation." In *Industrial Mechatronics and Automation (ICIMA), 2010 2nd International Conference on*, Vol. 2, pp. 240-243. IEEE, 2010.

[16] Shu-Ching Wang, Kuo-Qin Yan, Wen-Pin Liao, and Shun-Sheng Wang. "Towards a Load Balancing in a Three-Level Cloud Computing Network." In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, vol. 1, pp. 108-113. IEEE, 2010.

[17] Nae, Vlad, RaduProdan, and Thomas Fahringer. "Cost-efficient Hosting and Load Balancing of Massively Multiplayer Online Games." In *Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on*, pp. 9-16. IEEE, 2010.

[18] Stanojević, Rade, and Robert Shorten. "Load Balancing Vs. Distributed Rate Limiting: An Unifying Framework for Cloud Control." In *Communications, 2009. ICC'09. IEEE International Conference on*, pp. 1-6. IEEE, 2009.

[19] Zhao, Yi, and Wenlong Huang. "Adaptive Distributed Load Balancing Algorithm Based on Live Migration of Virtual Machines in Cloud." In *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, pp. 170-175. IEEE, 2009.

[20] Aameek Singh, Madhukar Korupolu, and Dushmanta Mohapatra. "Server-storage Virtualization: Integration and Load Balancing in Data Centers." In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, p. 53. IEEE Press, 2008.

[21] Kansal, Nidhi Jain, and Inderveer Chana. "Cloud Load Balancing Techniques: A Step Towards Green Computing." *IJCSI International Journal of Computer Science Issues* 9, No. 1 (2012): 238-246.

[22] Ram Prasad Padhy. "Load Balancing in Cloud Computing Systems." PhD diss., National Institute of Technology, Rourkela, 2011.

[23] Devine, Karen D., Erik G. Boman, Robert T. Heaphy, Bruce A. Hendrickson, James D. Teresco, Jamal Faik, Joseph E. Flaherty, and Luis G. Gervasio. "New Challenges in Dynamic Load Balancing." *Applied Numerical Mathematics* 52, No. 2 (2005): 133-152.

[24] Williams, Roy D. "Performance of Dynamic Load Balancing Algorithms for Unstructured Mesh Calculations." *Concurrency: Practice and Experience* 3, No. 5 (1991): 457-481.

[25] Shreedhar, Madhavapeddi, and George Varghese. "Efficient Fair Queuing Using Deficit RoundRobin." *Networking, IEEE/ACM Transactions on* 4, no. 3 (1996): 375-385.

[26] Wu, Min-You, Wei Shu, and Hong Zhang. "Segmented Min-Min: A Static Mapping Algorithm for Meta-Tasks on Heterogeneous Computing Systems." In *hcw*, p. 375. IEEE, 2000.

[27] Muthucumaru Maheswaran, Shoukat Ali, H. J. Siegal, Debra Hensgen, and Richard F. Freund. "Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems." In *Heterogeneous Computing Workshop, 1999.(HCW'99) Proceedings. Eighth*, pp. 30-44. IEEE, 1999.

[28] Kokilavani, T., and Dr DI George Amalarethinam. "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing." *International Journal of Computer Applications* 20, No. 2 (2011): 43-49.

[29] Graham Ritchie, John Levine, ―‖A Fast Effective Local Search for Scheduling Independent Jobs in Heterogeneous Computing Environments‖ Center for Intelligent Systems and their applications School of Informatics University of Edinburg.

[30] MsNitika, MsShaveta, and Mr Gaurav Raj. "Comparative Analysis of Load Balancing Algorithms in Cloud Computing." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 1, No. 3 (2012): pp-120.

[31] Zhang, Qi, Lu Cheng, and RaoufBoutaba. "Cloud Computing: State-Of-The-Art and Research Challenges." *Journal of Internet Services and Applications* 1, No. 1 (2010): 7-18.

[32] Nitika, Shaveta, Gaurav Raj, International Journal of Advanced Research in Computer Engineering and Technology Vol-1 issue-3 May-2012.

[33] Zenon Chaczko, Venkatesh Mahadevan, Shahrzad Aslanazadeh, and Christopher, IPCSIT Vol14, IACSIT Press Singapore 2011.

[34] ShuChing Wang, Kuo-Qin Yan Wen-pin Liao, and Shun Sheng Wang Chaoyang University of Technology, Taiwan R.O.C.

[35] Che-Lun Hung, Hsiao-hsi Wang, and Yu- ChenHu ―Efficient Load Balancing Algorithm for Cloud Computing Network‖.

[36] Yatendrasahu, M. K. Pateriya ―Cloud Computing Overview and Load Balancing Algorithms‖, Internal Journal of Computer Application Vol-65 No.24, 2013.

[37] NayandeepSran, Navdeepkaur ―Comparative Analysis of Existing Load Balancing Techniques in Cloud Computing‖, International Journal of Engineering Science Invention, Vol-2 Issue-1 2013.

[38] Fazel Mohammadi, Dr. Shahram Jamali and Masoud Bekravi ―Survey on Job Scheduling Algorithms in Cloud Computing‖ International Journal of Emerging Trends & Technology in Computer Science (IJETTCS) Volume 3, Issue 2, March – April 2014.

[39] Baris Yuce, Michael S. Packianather, Ernesto Mastrocinque, Duc Truong Pham and Alfredo Lambiase ―Honey Bees Inspired Optimization Method: The Bees Algorithm‖ insects 1 July 2013; Published: 6 November 2013.

[40] BibhudattaSahoo, SudiptaMohapatra, and Sanjay Kumar Jena ―A Genetic Algorithm Based Dynamic Load Balancing Scheme for Heterogeneous Distributed Systems‖ Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA 2008, Las Vegas, Nevada, USA, July 14-17, 2008,2 Volumes. CSREA Press 2008.

[41] Kousik Dasguptaa, Brototi Mandalb, Paramartha Duttac, Jyotsna Kumar Mondald, Santanu Dame, A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing‖ International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) Vol 10, 2013.

[42] Shilpa V Pius, Shilpa T S ―Survey on Load Balancing in Cloud Computing‖ International Conference on Computing, Communication and Energy Systems (ICCCES-2014).

[43] J. Uma, V. Ramasamy, A. Kaleeswaran ― Load Balancing Algorithms in Cloud Computing Environment - A Methodical Comparison‖ International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 3, Issue 2, February 2014.

[44] Brototi Mondala, Kousik Dasgupta, Paramartha Dutta ―Load Balancing in Cloud Computing using Stochastic Hill Climbing - A Soft Computing Approach‖ Procedia Technology 4 (2012) 783 –789 2212-0173 © 2012 Published by Elsevier Ltd. doi: 10.1016/j.protcy.2012.05.128 C3IT-2012.

[45] Albert Y. Zomaya, Senior Member, IEEE, and Yee-Hwei The ―Observations on Using Genetic Algorithms for Dynamic Load Balancing‖ IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 9, September 2001.

[46] N. S. Raghava and Deepti Singh ―Comparative Study on Load Balancing Techniques in Cloud Computing‖ Open Journal of Mobile Computing and Cloud Computing Vol.1, No.1, Aug. 2014.

[47] Bharti Suri ―Implementing Ant Colony Optimization for Test Case Selection and Prioritization‖ Bharti Sure et al. / International Journal on Computer Science and Engineering (IJCSE).

[48] Shiny ―Load Balancing In Cloud Computing: A Review‖ IOSR Journal of Computer Engineering (IOSR -JCE) e-ISSN: 2278-0661, p- ISSN: 2278-8727 Vol.15, Issue 2 (Nov. - Dec. 2013), pp 22-29.

[49] Mohiuddin Ahmed, Abu Sina Md. Raju Chowdhury, Mustaq Ahmed, Md. Mahmudul Hasan Rafee ―An Advanced Survey on Cloud Computing and State-of-the-art Research Issues‖ IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 1, January 2012.

[50] Tushar Desai, JigneshPrajapati, A Survey of Various Load Balancing Techniques and Challenges in Cloud Computing, International Journal of Scientific & Technology Research. Vol.2 Iss.11. Nov.2013 pp:158 -161.

[51] O M Elzeki, M Z Reshad, Improved Max-Min Algorithm in Cloud Computing, International Journal of Computer Applications. Vol. 50, Iss. 12. July 2012. PP: 22-27.

[52] Zenon Chaczko, Venkatesh Mahadevan, Shahrzad Aslanzadeh, Christopher Mcdermid (2011) Availabity and Load Balancing in Cloud Computing‖ International Conference on Computer and Software Modeling IPCSIT Vol.14 IACSIT Press, Singapore 2011.

[53] Ranjan Kumar Mondal, Enakshmi Nandi, and Debabrata Sarddar. "Load Balancing Scheduling with Shortest Load First." *International Journal of Grid and Distributed Computing* 8.4 (2015): 171-178.

[54] Ranjan Kumar Mondal, Debabrata Sarddar ―Load Balancing with Task Subtraction of Same Nodes‖. International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online) Vol. 3, Issue 4, pp: (162-166), Month: October - December 2015.

[55] Ms Jayshri Damodar Pagare and Dr. Nitin A Koli, Performance Analysis of an Energy Efficient Virtual Machine Consolidation Algorithm in Cloud Computing, International Journal of Computer Engineering and Technology (IJCET), Volume 6, Issue 5, May (2015), pp. 24-35

[56] S. Saran Raj and R. Hariharan, Person Anamnesis Tracking System Using Cloud Computing, International Journal of Civil Engineering and Technology, 8(10), 2017, pp. 271–278.

[57] Smita N. Gambhire, Asha P. Gavhane and Mahadev Patil, Power of Cloud Computing in Customer Relationship Management, International Journal of Management (IJM), Volume 3, Issue 1, January- April (2012), pp. 225-230

[58] Rishi Aluri, Shriya Mehra, Apoorva Sawant, Pankti Agrawal and Mayank Sohani, Priority based Non-Preemptive Shortest Job First Resource Allocation Technique in Cloud Computing. International Journal of Computer Engineering & Technology, 9(2), 2018, pp. 132–139.

[59] D. Pratiba and Dr. G. Shobha, Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing, International Journal of Computer Engineering and Technology (IJCET), Volume 4, Issue 3, May-June (2013), pp. 441-448

[60] Prof. Supriya Santosh. Wagh, Prospect of Adopting Cloud Computing in Indian Manufacturing Industry, International Journal of Management (IJM), Volume 6, Issue 6, June (2015), pp. 48-56

[61] G.Ashok kumar and P.Srinivasulu, Protection of Data Using Linear Programming and FHE Technique in Cloud Computing, International Journal of Information Technology & Management Information System (IJITMIS), Volume 4, Issue 1, January – April (2013), pp. 24-30