World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(REVIEW ARTICLE)

Optimized load balancing technique in cloud computing systems

Nagaraj. M [1,*], Trimuke Digambar [2] and Ameena Firdous Nikhat [1]

[1] *Department of Computer Science and Engineering, Government Polytechnic., kalaburgi, Karnataka, India.*
[2] *Department of Computer Science and Engineering, Government Women's Polytechnic Kalaburagi Karnataka, India.*

World Journal of Advanced Research and Reviews, 2019, 03(02), 111–120

Publication history: Received on 20 August 2019; revised on 23 September 2019; accepted on 26 September 2019

Article DOI: https://doi.org/10.30574/wjarr.2019.3.2.0050

Abstract

Cloud computing" is a term, which involves virtualization, distributed computing, Networking, software and web services. A cloud consists of several elements such As clients, datacenter and distributed servers. It includes fault tolerance, high availability, Scalability, flexibility, reduced overhead for users, reduced cost of ownership, on Demand services etc. Central to these issues lies the establishment of an effective load balancing algorithm. The load can be CPU load, memory capacity, delay or network load. Load Balancing is the process of distributing the load among various nodes of a distributed System to improve both resource utilization and job response time while also avoiding A situation where some of the nodes are heavily loaded while other nodes are idle or Doing very little work. Load balancing ensures that all the processor in the system or Every node in the network does approximately the equal amount of work at any instant of time. This technique can be sender initiated, receiver initiated or symmetric type (Combination of sender initiated and receiver initiated types).

Keywords: Cloud computing; Load balancing; Resource allocation; Optimization algorithms; Distributed systems Scalability

1. Introduction

Cloud computing is an on demand service in which shared resources, information, software and other devices are provided according to the clients requirement at specific time. It's a term which is generally used in case of Internet. The whole Internet can be viewed as a cloud. Capital and operational costs can be cut using cloud computing.

Load balancing in cloud computing systems is really a challenge now. Always a distributed solution is required. Because it is not always practically feasible or cost efficient to maintain one or more idle services just as to fulfill the required demands. Jobs can't be assigned to appropriate servers and clients individually for efficient load balancing as cloud is a very complex structure and components are present throughout a wide spread area. Here some uncertainty is attached while jobs are assigned[1].

As the whole Internet can be viewed as a cloud of many connection-less and connection oriented services, thus concept of load balancing in Wireless sensor networks (WSN) proposed in can also be applied to cloud computing systems as WSN is analogous to a cloud having no. of master computers (Servers) and no. of slave computers (Clients) joined in a complex structure. A comparative study of different algorithms has been carried out using divisible load scheduling .

* Corresponding author: Nagaraj. M

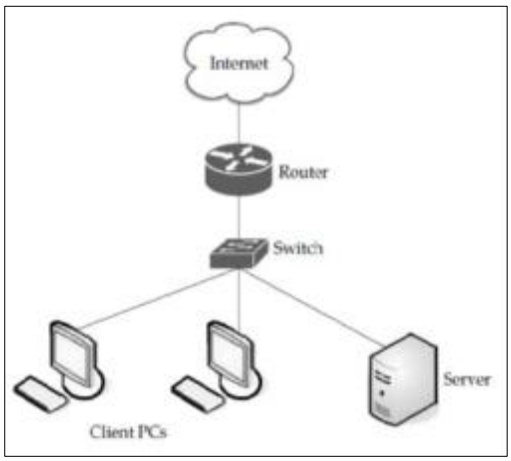

Figure 1 A cloud is used in network diagrams to depict the Internet

In case of Cloud computing services can be used from diverse and widespread resources, rather than remote servers or local machines. There is no standard definition of Cloud computing. Generally it consists of a bunch of distributed servers known as masters, providing demanded services and resources to different clients known as clients in a network with scalability and reliability of data center. The distributed computers provide on-demand services.

- Services may be of software resources (e.g. Software as a Service, SaaS) or
- physical resources (e.g. Platform as a Service, PaaS) or
- Hardware/Infrastructure (e.g. Hardware as a Service, HaaS or Infrastructure as a Service, IaaS). Amazon EC2 (Amazon Elastic Compute Cloud) is an example of cloud computing services.

A Cloud system consists of 3 major components such as clients, datacenter, and Distributed servers. Each element has a definite purpose and plays a specific role[2].

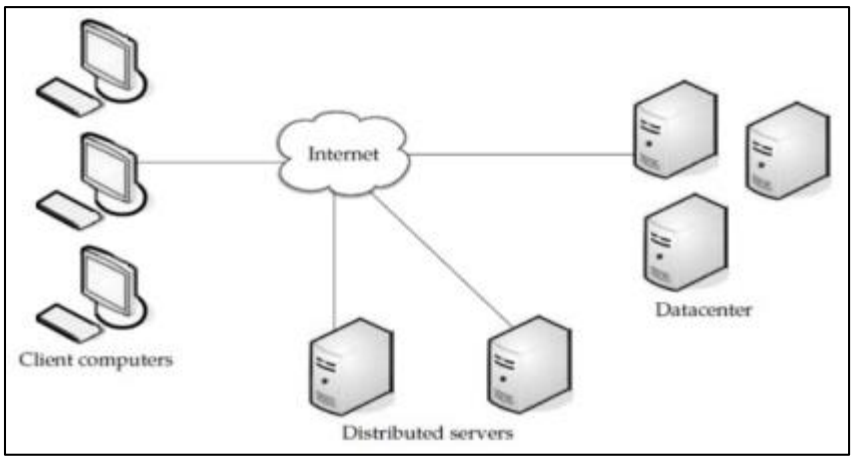

Figure 2 Cloud Components

End users interact with the clients to manage information related to the cloud. Clients Generally fall into three categories as given in [1]:

- Mobile: Windows Mobile Smartphone, smart phones, like a Blackberry, or an iPhone.
- Thin: They don't do any computation work. They only display the information. Servers do all the works for them. Thin clients don't have any internal memory.
- Thick: These use different browsers like IE or Mozilla Firefox or Google Chrome to connect to the Internet cloud.

Now-a-days thin clients are more popular as compared to other clients because of Their low price, security, low consumption of power, less noise, easily replaceable and repairable, etc[3-5].

Data center is nothing but a collection of servers hosting different applications. An end User connects to the data center to subscribe different applications. A data center may exist at a large distance from the clients. Now-a-days a concept called virtualization is used to install a software that allow Multiple instances of virtual server applications.

Distributed servers are the parts of a cloud which are present throughout the Internet Hosting different applications. But while using the application from the cloud, the user will feel that he is using this application from its own machine.

Types of Clouds: Based on the domain or environment in which clouds are used, clouds can be divided Into 3 categories:

- Public Clouds
- Private Clouds
- Hybrid Clouds (combination of both private and public clouds)

It is a very useful concept in context of cloud systems. Virtualization means"something Which isn't real", but gives all the facilities of a real. It is the software implementation of a computer which will execute different programs like a real machine.Virtualization is related to cloud, because using virtualization an end user can use different services of a cloud. The remote data center will provide different services in a full or partial virtualized manner.

2 types of virtualization are found in case of clouds as given in [1]:

- Full virtualization
- Para virtualization

In case of full virtualization a complete installation of one machine is done on the Another machine. It will result in a virtual machine which will have all the software's that are present in the actual server.

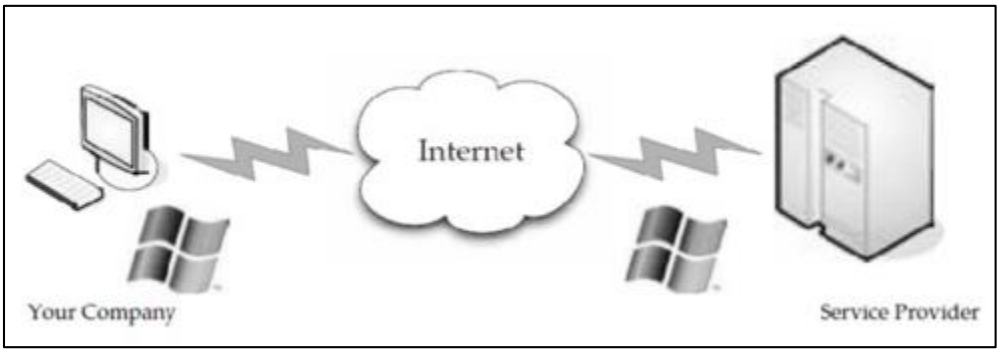

Figure 3 Full Virtualization

Here the remote data center delivers the services in a fully virtualized manner. Full Virtualization has been successful for several purposes as pointed out in [1]:

- Sharing a computer system among multiple users
- Isolating users from each other and from the control program
- Emulating hardware on another machine
- Para virtualization

In Para virtualization, the hardware allows multiple operating systems to run on single Machine by efficient use of system resources such as memory and processor. E.g. VMware software. Here all the services are not fully available, rather the services are provided partially[6-8].

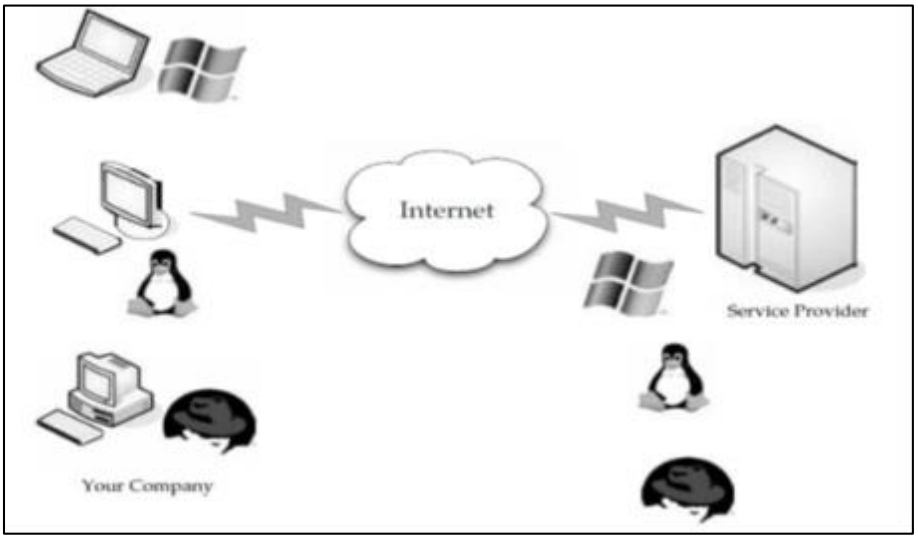

Figure 4 Para virtualization (adopted from [1])

Para virtualization has the following advantages as given in [1]:

- Disaster recovery: In the event of a system failure, guest instances are moved to another hardware until the machine is repaired or replaced.
- Migration: As the hardware can be replaced easily, hence migrating or moving the different parts of a new machine is faster and easier.
- Capacity management: In a virtualized environment, it is easier and faster to add more hard drive capacity and processing power. As the system parts or Hardware's can be moved or replaced or repaired easily, capacity management is simple and easier.

Goals of Load balancing are:

- To improve the performance substantially
- To have a backup plan in case the system fails even partially
- To maintain the system stability
- To accommodate future modification in the system

2. Literature Survey

Load balancing is a process of reassigning the total load to the individual nodes of the collectiveSystem to make resource utilization effective and to improve the response time of the job, simultaneously removing a condition in which some of the nodes are over loaded while some others are under loaded. A load balancing algorithm which is dynamic in nature does-not consider the previous state or behavior of the system, that is, it depends on the present-behavior of the system. The important things to consider while developing such algorithm-are : estimation of load, comparison of load, stability of different system, performance of-system, interaction between the nodes, nature of work to be transferred, selecting of nodes-and many other ones. This load considered can be in terms of CPU load, amount of-memory used, delay or Network load.

A Novel Load Balancing Algorithm for Dynamically Scalable Web Services is Proposed by Yi Lua , QiaominXiea , Gabriel Kliotb , Alan Gellerb , James R. Larusb , Albert Greenberg[1] They proposed an algorithm for novel class of algorithms called Join-Idle-Queue (JIQ) for distributed load balancing in large systems. Unlike algorithms such as Power-of-Two, the JIQ algorithm incurs no communication overhead between the dispatchers and processors at job arrivals. We analyze the JIQ algorithm in the large system limit and find that it effectively results in a reduced system load, which produces 30-fold reduction in queueing overhead compared to Power-of-Two at medium to high load. An extension of the basic JIQ algorithm deals with very high loads using only local information of server load.

Load Balancing for Future Internet: An Approach Based on Game Theory ,Shaoyi Song, TingjieLv, and Xia Chen[2] In this paper, they introduce a load balancing model for future internet. We formulate the static load balancing problem in the

model proposed above as non-cooperative game among users and cooperative game among processors. Based on this model, we derive a load balancing algorithm for computing center. Finally, we execute the algorithm presented in this paper with another three algorithms for comparison purpose. The advantages of our algorithm are better scalability to the model, improving system performance, and low cost on maintaining system information.

Optimum Load Balancing of Cloud-lets Using Honey Bee Behavior Load Balancing Algorithm, Obaid Bin Hassan, A Sarfaraz Ahmad[3] Cloud Computing enables to compute resources as a utility and consumption of computing resources. In Cloud Computing we deploy groups of remote servers and software networks that allow centralized data storage and online access to computer services or resources. Load balancing has become one of the key issues with the rapid growth of web traffic and the services available under cloud environments. It is the job of Load Balancer to distribute the load among different virtual machines so that all the nodes get equally loaded. To attain the maximum throughput and minimum time various researchers throughout the world proposed many algorithms and approaches. In this paper we propose a solution for the existing load balancing algorithm "Honey Bee Behavior Inspired Load Balancing Algorithm" to provide optimum balancing of tasks in cloud environments. Uma Singhal and Sanjeev Jain[4] In this paper, they survey a special group of Swarm intelligence based load balancing algorithm and discuss the advantages and issues of these algorithms for cloud computing environment.

3. Types of Load balancing algorithms

Depending on who initiated the process, load balancing algorithms can be of three Categories as given in [4]:

- Sender Initiated: If the load balancing algorithm is initialized by the sender
- Receiver Initiated: If the load balancing algorithm is initiated by the receiver

3.1. Dynamic Load balancing algorithm

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 as given in [4]:

- Static: It doesn't depend on the current state of the system. Prior knowledge of the system is needed
- Dynamic: Decisions on load balancing are based on current state of the system. No prior knowledge is needed. So it is better than static approach.

3.2. Dynamic Load balancing algorithm

In a distributed system, dynamic load balancing can be done in two different ways:

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

- Cooperative and
- Non-cooperative [4].

In the first one, the nodes work side-by-side to achieve a common objective, for example, to improve the overall response time, etc. In the second form, each node works independently toward a goal local to it, for example, to improve the response time of a local task[9].

Dynamic load balancing algorithms of distributed nature, usually generate more messages than the non-distributed ones because, each of the nodes in the system needs to interact with every other node. A benefit, of this is that even if one or more nodes in the system fail, it will not cause the total load balancing process to halt; it instead would affect the system performance to some extent. Distributed dynamic load balancing can introduce immense stress on a system in which each node needs to interchange status information with every other node in the system. It is more advantageous when most of the nodes act individually with very few interactions with others.

In non-distributed type, either one node or a group of nodes do the task of load Balancing. Non-distributed dynamic load balancing algorithms can take two forms:

- Centralized and
- Semi-distributed.

In the first form, the load balancing algorithm is executed only by a single node in the whole system: the central node. This node is solely responsible for load balancing of the whole system. The other nodes interact only with the central node. In semi-distributed form, nodes of the system are partitioned into clusters, where the load balancing in each cluster is of centralized form. A central node is elected in each cluster by appropriate election technique which takes care of load balancing within that cluster. Hence, the load balancing of the whole system is done via the central nodes of each cluster[4].

Centralized dynamic load balancing takes fewer messages to reach a decision, as the Number of overall interactions in the system decreases drastically as compared to the semi distributed case. However, centralized algorithms can cause a bottleneck in the system at the central node and also the load balancing process is rendered useless once the central node crashes. Therefore, this algorithm is most suited for networks with small size. Policies or Strategies in dynamic load balancing There are 4 policies[10-12]

- Transfer Policy: The part of the dynamic load balancing algorithm which selects A job for transferring from a local node to a remote node is referred to as Transfer policy or Transfer strategy.
- Selection Policy: It specifies the processors involved in the load exchange (processor matching)
- Location Policy: The part of the load balancing algorithm which selects a destination node for a transferred task is referred to as location policy or Location strategy.
- Information Policy: The part of the dynamic load balancing algorithm responsible For collecting information about the nodes in the system is referred to as Information policy or Information strategy.

3.3. Distributed Load Balancing for the Clouds

In complex and large systems, there is a tremendous need for load balancing. For Simplifying load balancing globally (e.g. in a cloud), one thing which can be done is, Employing techniques would act at the components of the clouds in such a way that the Load of the whole cloud is balanced. For this purpose, we are discussing three types of Solutions which can be applied to a distributed system [7]: honeybee foraging algorithm, a biased random sampling on a random walk procedure and Active Clustering.

3.4. Honeybee Foraging Algorithm

This algorithm is derived from the behavior of honey bees for finding and reaping Food. There is a class of bees called the forager bees which forage for food sources, upon finding one, they come back to the beehive to advertise this using a dance called waggle dance. The display of this dance, gives the idea of the quality or quantity of food and also its distance from the beehive. Scout bees then follow the foragers to the location of food and then began to reap it. They then return to the beehive and do a waggle dance, which gives an idea of how much food is left and hence results in more exploitation or abandonment of the food source.

In case of load balancing, as the web servers demand increases or decreases, the services are assigned dynamically to regulate the changing demands of the user. The servers are grouped under virtual servers (VS), each VS having its own virtual service queues. Each server processing a request from its queue calculates a profit or reward, which is analogous to the quality that the bees show in their waggle dance. One measure of this reward can be the amount of time that the CPU spends on the processing of a request. The dance floor in case of honey bees is analogous to an advert board here. This board is also used to advertise the profit of the entire colony.Each of the servers takes the role of either a forager or a scout. The server after

Processing a request can post their profit on the advert boards with a probability of **Pr. A** Server can choose a queue of a VS by a probability of px showing forage/explore behavior,or it can check for advertisements (see dance) and serve it, thus showing scout behavior. A server serving a request, calculates its profit and compare it with the colony profit and then sets its px. If this profit was high, then the server stays at the current virtual server; posting an advertisement for it by probability pr. If it was low, then the server returns to the forage or scout behavior.

3.5. Biased Random Sampling

Here a virtual graph is constructed, with the connectivity of each node (a server is

Treated as a node) representing the load on the server. Each server is symbolized as a node in the graph, with each in degree directed to the free resources of the server.

Regarding job execution and completion,

- Whenever a node does or executes a job, it deletes an incoming edge, which indicates reduction in the availability of free resource.
- After completion of a job, the node creates an incoming edge, which indicates an Increase in the availability of free resource. The addition and deletion of processes is done by the process of random sampling.

The walk starts at any one node and at every step a neighbor is chosen randomly. The Last node is selected for allocation for load. Alternatively, another method can be used for selection of a node for load allocation, that being selecting a node based on certain criteria like computing efficiency, etc. Yet another method can be selecting that node for load allocation which is under loadedi.e. having highest in degree. If b is the walk length, then, as b increases, the efficiency of load allocation increases. We define a threshold value of b, which is generally equal to log n experimentally.

A node upon receiving a job, will execute it only if its current walk length is equal To or greater than the threshold value. Else, the walk length of the job under consideration is incremented and another neighbor node is selected randomly. When, a job is executed by a node then in the graph, an incoming edge of that node is deleted. After completion of the job, an edge is created from the node initiating the load allocation process to the node which was executing the job. Finally what we get is a directed graph. The load balancing scheme used here is fully Decentralized, thus making it apt for large network systems like that in a cloud[13-15].

3.6. Active Clustering

Active Clustering works on the principle of grouping similar nodes together and Working on these groups. The process involved is:

- A node initiates the process and selects another node called the matchmaker node from its neighbors satisfying the criteria that it should be of a different type than the former one.
- The so called matchmaker node then forms a connection between neighbors of it which is of the same type as the initial node.
- The matchmaker node then detaches the connection between itself and the initial node.

The above set of processes is followed iteratively. The time required for completing a task within one process is very high. So the task Is divided into no. of sub-tasks and each sub-task is given one one job. Let the task S is divided into no. of sub-tasks S1,S2,S3...Sn. Out of these some are executed sequentially and some are executed parallel. So the total time period for completing the task decreases and hence the performance increases. These sub-tasks can be represented in a graph structure known as state diagram. An example is given below.

S1 is executed first. S2,S3,S4 and S5 can be executed parallel during the same time Slice. S18 requires the execution of S6 and S7 both, but S19 requires the execution of S8 And so on for all the sub tasks as shown in the state diagram. Our aim is to execute these Tasks in different nodes of a distributed network so that the performance can be enhanced. The distributed network may follow different topologies. The tasks are distributed Over the whole network. One topological network connects with the other through a gateway. One of the physical topologies forming a cloud is shown in the diagram 6. This distributed network is a cloud, because some of the nodes are Mobile clients, Some of them are Thin and some are Thick clients. Some of them are treated as masters and some are treated as slaves. There are one or more data centers distributed among the various nodes, which keeps track of various computational details. Our aim is to apply the Divisible Load Scheduling Theory (DLT) proposed in [9] for the clouds of different sizes and analyze different performance parameters for different algorithms under DLT and compare them[16].

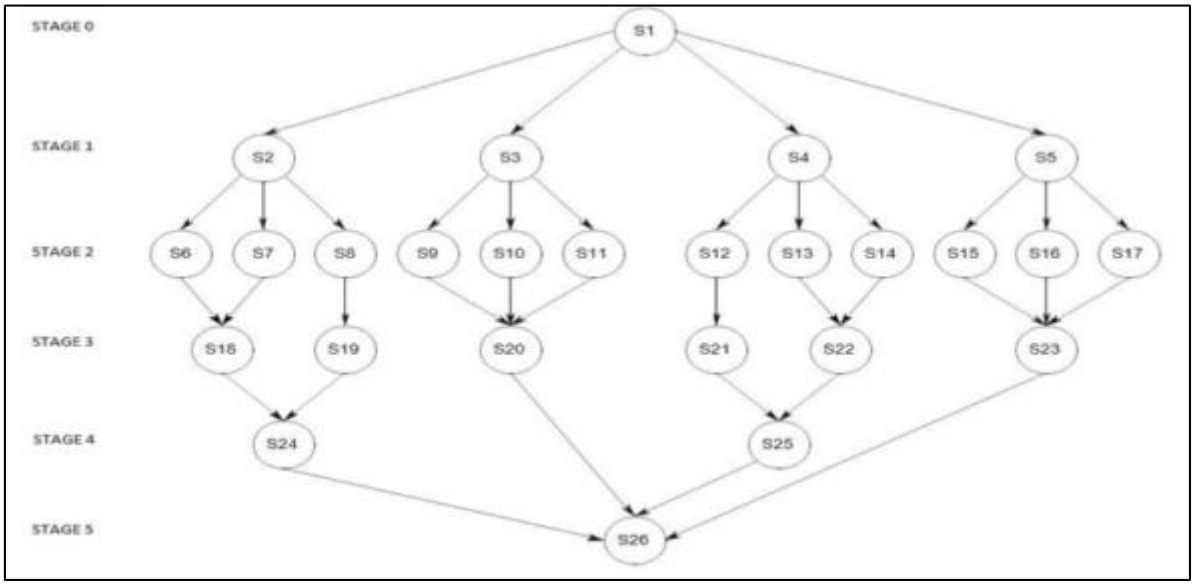

Figure 5 State Diagram

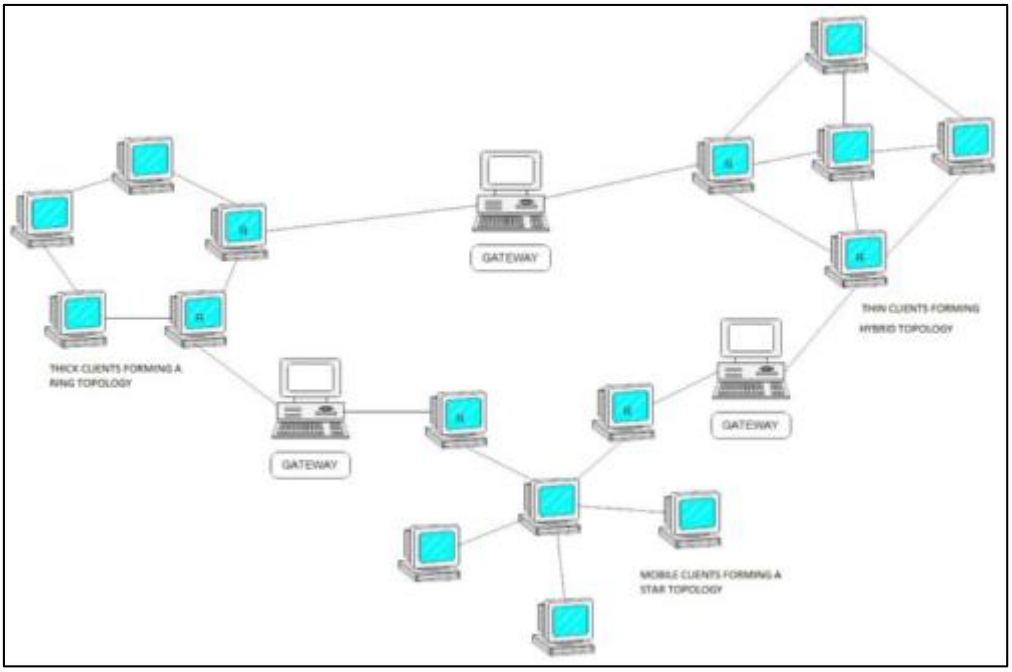

Figure 6 A cloud showing different topologies

4. Divisible Load Scheduling Theory in Clouds

Divisible load scheduling theory (DLT) in case of clouds is an optimal division of Loads among a number of master computers, slave computers and their communication Links. Our objective is to obtain a minimal partition of the processing load of a cloud connected via different communication links such that the entire load can be distributed and processed in the shortest possible amount of time. The whole Internet can be viewed as a cloud of many connection-less and connection oriented services. The concept of load balancing in Wireless sensor networks (WSN) proposed in [9] can also be applied to clouds as WSN is analogous to a cloud having no. of master computers (Servers) and no. of slave computers (Clients). The slave computers are assumed to have a certain measurement capacity. We assume that computation will be done by the master computers, once all the measured data is gathered from corresponding slave computers. Only the measurement and communication times of the slave computers are considered and the computation time of the slave computers is neglected. Here we consider both heterogeneous and homogeneous clouds.

That is the cloud elements may possess different measurement capacities, and communication link speeds or the same measurement capacities, and communication link speeds. One slave computer may be connected to one or more master computers at a certain instant of time. In DLT in case of clouds, an arbitrarily divisible load without having any previous Relations is divided and first distributed among the various master computers (for simplicity here the load is divided equally between the master computers) and the each master computer distributes the load among the corresponding slave computers so that the entire load can be processed in shortest possible amount of time. An important reason for using DLT is its flexibility, tractability, data parallelism, computational difficulties [9].

5. Conclusion

Load balancing is crucial for optimizing performance and efficiency in cloud computing. Modern approaches focus on dynamic, adaptive techniques that use real-time data to make intelligent routing decisions. Machine learning and AI show promise in predicting workload patterns and optimizing resource allocation. Hybrid approaches combining multiple algorithms offer flexibility for diverse scenarios. Key areas for improvement include enhancing energy efficiency, improving scalability, minimizing latency, increasing fault tolerance, and optimizing for heterogeneous hardware. As cloud computing evolves, load balancing must adapt to new architectures and workloads. The most effective techniques balance multiple objectives, including response time, resource utilization, and cost. They consider factors such as network conditions, server capacities, and application requirements. Continuous monitoring and adjustment are essential to maintain optimal performance in dynamic cloud environments.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Yi Lua , QiaominXiea , Gabriel Kliotb , Alan Gellerb , James R. Larusb , Albert Greenberg,A Novel Load Balancing Algorithm for Dynamically Scalable Web Services.

[2] Shaoyi Song, TingjieLv, and Xia Chen, Load Balancing for Future Internet: An Approach Based on Game Theory.

[3] Obaid Bin Hassan, A Sarfaraz Ahmad, Optimum Load Balancing of Cloud-lets Using Honey Bee Behavior Load BalancingAlgorithm.

[4] Uma Singhal and Sanjeev Jain, Honey Bee Behavior Inspired Load Balancing Algorithm to provide optimum balancing of tasks in cloud environments.

[5] Anthony T.Velte, Toby J.Velte, Robert Elsenpeter, Cloud Computing A Practical Approach, TATA McGraw-HILL Edition 2010.

[6] Martin Randles, David Lamb, A. Taleb-Bendiab, A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing, 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops.

[7] Mladen A. Vouk, Cloud Computing Issues, Research and Implementations, Proceedings of the ITI 2008 30th Int. Conf. on Information Technology Interfaces, 2008, June 23-26.

[8] Ali M. Alakeel, A Guide to Dynamic Load Balancing in Distributed Computer Systems, IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.6, June 2010.

[9] http://www-03.ibm.com/press/us/en/pressrelease/22613.wss

[10] http://www.amazon.com/gp/browse.html?node=201590011

[11] Martin Randles, EnasOdat, David Lamb, Osama Abu- Rahmeh and A. Taleb-Bendiab, A Comparative Experiment in Distributed Load Balancing, 2009 Second International Conference on Developments in eSystems Engineering.

[12] Peter S. Pacheco, Parallel Programming with MPI, Morgan Kaufmann Publishers Edition 2008

[13] MequanintMoges, Thomas G.Robertazzi, Wireless Sensor Networks: Scheduling for Measurement and Data Reporting, August 31, 2005

[14] Suneel K S, Ravichandra A J. ,Dr. H S GuruprasadIJESRT Enhanced Load Balancing Algorithm in Three-Tier Cloud Computing December 2014

[15] Punit Gupta and Satya Prakash Ghrera Load and Fault Aware Honey Bee Scheduling Algorithm for Cloud InfrastructureSpringerInternational Publishing Switzerland 2015

[16] Uma Singhal and Sanjeev Jain, International Journal of Hybrid Information Technology Vol.8, No.1 (2015) An Analysis of Swarm Intelligence based Load Balancing Algorithms in a Cloud Computing Environment