A Thesis/Project/Dissertation Report

on

"Prediction model in Machine Learning for Resource Allocation for Sustainable and Reliable Cloud Computing Environment "

Submitted in partial fulfillment of the requirement for the award of the degree of

B.Tech in Computer Science and Engineering

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of
Name of Supervisor: Mr. M. Arvindhan Sir
Designation: Assistant Professor

Submitted By
Name of Student/s: Shalini Sinha,
Shashank Bora
Enrollment/Admission No.: 19SCSE1010390,
19SCSE1010670

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

DECEMBER, 2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"PREDICTION MODEL IN MACHINE LEARNING FOR RESOURCE ALLOCATION FOR SUSTAINABLE AND RELIABLE CLOUD COMPUTING ENVIRONMENT "** in partial fulfillment of the requirements for the award of the **B.Tech in Computer Science and Engineering** submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of July, 2021 to December, 2021, under the supervision of Mr. M. Arvindhan, Assistant Professor, Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering, Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

SHALINI SINHA, 19SCSE1010390
SHASHANK BORA, 19SCSE1010670

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor Name
Designation

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of **Shalini Sinha, 19SCSE1010390; Shashank Bora, 19SCSE1010670** has been held on __________________ and his/her work is recommended for the award of **B.Tech in Computer Science and Engineering.**

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: December, 2021
Place: Greater Noida

ACKNOWLEDGMENT

On this assignment, we gave it our best shot. This would not have been possible without the assistance and goodwill of a large number of people and organisations. We want to show our thanks to everyone. We are grateful to Mr. M. Arvindhan Sir for his direction and regular monitoring, as well as for supplying us with the essential project information and his help in seeing the project through to completion. We'd like to thank a member of the Galgotias University community for his exceptional cooperation and support in helping us complete this project. We would want to express our gratitude to the people in the business for their time and attention in this way. Our gratitude goes out to all of our project partners as well as those who volunteered their time and expertise to assist us.

ABSTRACT

The current cloud computing environment applies to almost every aspect of our life, thanks to easy access to the Internet. loud computing is a technology that uses the Internet to store and manage data on remote servers, then access that data over the Internet. Access to computer program resources, especially data storage (cloud storage) and computer power, without direct user-friendly management is necessary. Machine learning is a branch of artificial intelligence (AI) and computer science that focuses on using data and algorithms to mimic human learning and improve its accuracy. The rise in daily Internet traffic necessitates the acceptance of load balancing to make the vast majority of resources available in the loud. Implementing load balancing simply entails dividing incoming application load across existing application nodes. Too many complex statistics and formulas have been employed in order to achieve better resource utilisation and development. The unsupervised algorithm will provide the greatest solution for load balancing in a cloud environment in all of this complexity. Unsupervised algorithms are a subclass of machine learning models in which testing methods for previously learned data can be implemented. Machine learning algorithms provide us with a variety of analytical techniques as well as the ability to integrate labelled data sets. These algorithms detect hidden patterns or data collections without the need for human intervention. We shall now employ the unsupervised algorithm, K- Means clustering as the base algorithm and suggest improved version load balancing.

KEYWORDS

Virtualization, cloud compution, Task scheduling, Load balancing, Resource optimization, algorithms

Contents

INTRODUCTION

The modern world is becoming increasingly reliant on information networks. It is expected that both the rapid development of the Internet and the services made available through it, as well as the widespread use of solutions for information processing in all spheres of life, from business to administration to private information, will occur. The widespread usage of IT frameworks imposes increasing more noteworthy demands on the quality of services that these frameworks provide. The efficacy of IT solutions is an important factor in determining their quality. This factor has a significant impact on the ease of use, and in extreme circumstances, it may result in the inability to utilise a given solution. This article focuses on the development of load balancing algorithms in multi-server frameworks, ranging from classic clusters to cluster frameworks.

The term "loud computing" refers to anything that involves offering hosted services via the internet. These These services are divided into three main categories: infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS).

Multicloud is a method of allocating workload among several computers or other resources over the network in order to achieve optimal resource utilisation, maximise throughput, reduce response time, and avoid overburdening. It represents a balancing of loads. This research paper proposed a algorithm that focuses on load balancing to reduce overload or underload on multi-clouds. Task Scheduling algorithms or technique in cloud computing. Efficient task scheduling methods should fulfil the needs of clients and improve resource utilisation in order to improve the overall performance of the cloud computing environment. It is the most effective way to achieve efficient resource sharing and use. In the realm of information technology, loud omputing is becoming more popular (IT). One of the most difficult challenges is balancing the load in cloud computing. Cloud computing allows a large number of clients to access scavenged, virtualized hardware and/or software infrastructure through the Internet. The goal of load balancing is to restrict resource use, which will encourage reductions in energy consumption and carbon emissions (with regard to the environment), necessitating the need for cloud computing. This will be accomplished by applying the logs method to the database and leveraging the logs of client activity to keep track of the data. The future scopia is to implement the cloud on the vast proportion and green computing.

The K-means clustering algorithm computes the centroids and iterates until it finds the best one. The data points are assigned to a cluster in such a manner in this algorithm that the total of the squared distances between the data points and the centroid is as little as possible.

PROBLEM FORMULATION:

With the quick development in innovation, there is a tremendous expansion of information in the internet for its productive administration and limiting the multiplication issues. Disseminated document framework assumes a critical part in the administration of distributed storage which is appropriated among the different servers. Commonly a portion of these servers get over-burden for taking care of customer solicitations and others re-principle inactive. Colossal number of customers demands on a specific stockpiling server may in-wrinkle the heap of the servers and will prompt lull of that server or discard the customer demands if not went to ideal. This situation debases the general framework execution and builds the reaction time.

Therefore, we have proposed an approach that balances the load of storage servers and effectively utilizes the server capabilities and resources.

CLOUD COMPUTING OVERVIEW

The supply of computer resources through the internet, such as storage, processing power, databases, networking, analytics, artificial intelligence, and software applications, is known as cloud computing (the cloud). Companies may get the computing assets they need when they need them by outsourcing these resources rather than purchasing and maintaining a physical, on-premise IT infrastructure. This allows for more flexible resources, faster innovation, and cost savings. A cloud migration is often linked to data and IT transformation for many businesses. The basic principles of CC are to direct user attention to distributed, parallel, and virtualized computing systems. CC may have a massive client base with a massive physical computer infrastructure because to virtualization. In the realm of developing CC, security is a big problem.

Characteristics of cloud computing

On-demand self service

Broad network access

Resource pooling

Rapid elasticity

Types of cloud deployments

There are three primary types of cloud deployments. Each has unique benefits and organizations often benefit from using more than one.

Public Cloud:

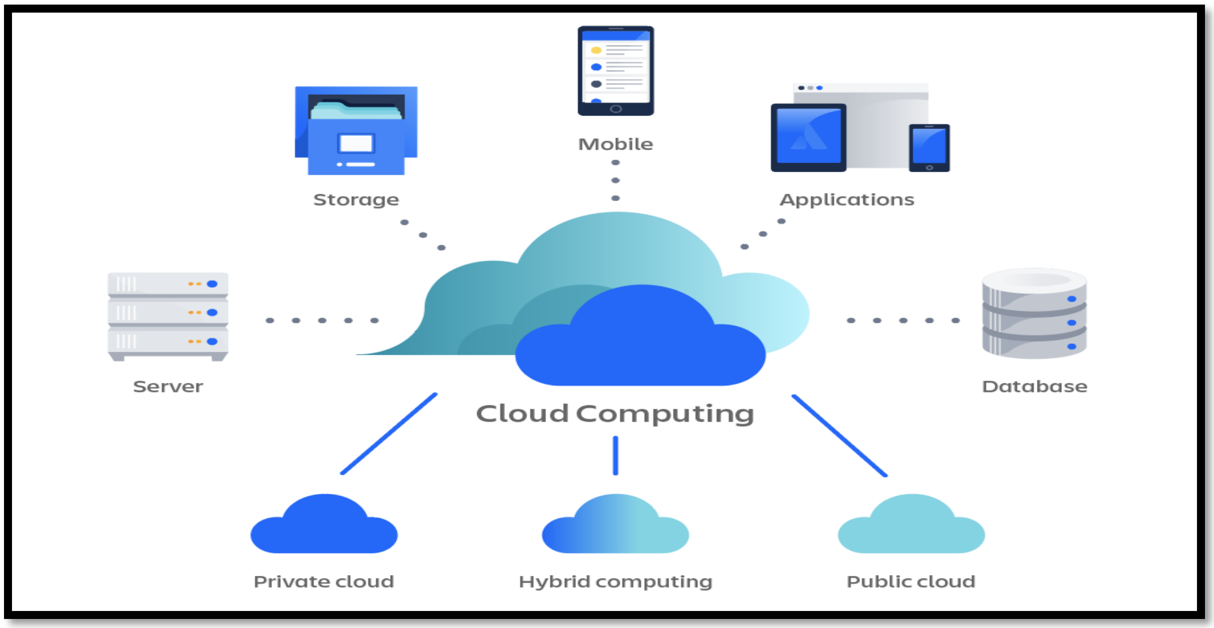
The public cloud exists for the benefit of the general public and as a viable alternative to the massive industry. The user has no access or insight to the computer infrastructure that the user is hosted on in this form of cloud.

Private Cloud:

When compared to public clouds, private clouds are more secure and costlier since they are maintained by an organisation or third party. Individual companies employ externally hosted private clouds managed by a third party that specialises in cloud infrastructure.

Hybrid Cloud:

This sort of cloud combines numerous clouds, each of which retains its own identity while being connected together as a single unit.



CLASSIFICATION OF CLOUDS BASED ON THE SERVICE PROVIDER

The clouds are mainly divided into three forms based on the various services offered by them. These three modes of cloud services include;

IaaS which entails the provisions of hardware associated services through utilization of CC principles. It offers virtual-machine, virtual storage, file or object storage, virtual infrastructure, IP addresses, disk image library raw block storage, virtual local area networks load balancer, firewalls, and software. Providers supplies them upon demand from the large tarns installed in the data center.

PaaS models involves the cloud provider providing a computing platform, which is inclusive of database, web server, operating system, and programming languageexecution atmosphere. Application developers operate their software solutions on the cloud platform without

necessitating the absorption of software and hardware layers or the cost of buying and handling them.

SaaS- entails providing the entire software in the cloud. Users on pay per use basis are granted access to the software application which is hosted by the cloud providers.

K-MEANS CLUSTERING

K-Means Clustering is a type of unsupervised machine learning method that divides an unlabeled dataset into groups. K indicates how many pre-defined clusters must be constructed during the procedure; for example, if K=2, two clusters will be created, if K=3, three clusters will be created, and so on. It allows us to divide data into separate groups and gives a straightforward method for determining group categories in an unlabeled dataset without any training. It's a centroid-based method,meaning that each cluster has its own centroid. The primary purpose of this technique is to lower the sum of distances between data points and clusters.

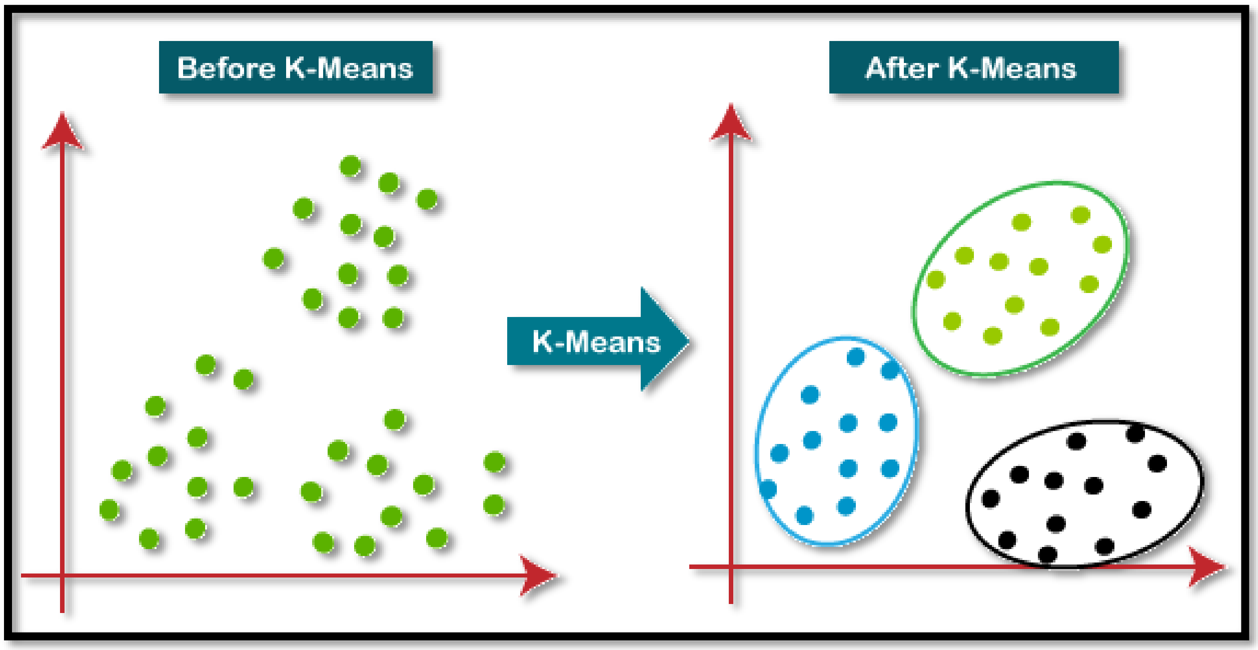
The k-means clustering technique has a number of advantages.

- It's simple to implement, yet it can handle large data sets.

- Ensures that everything is in sync.
- It may be used to warm up centroid placements.
- Quickly adapts to new conditions.
- Generalizes to different cluster shapes and sizes, including elliptical clusters.

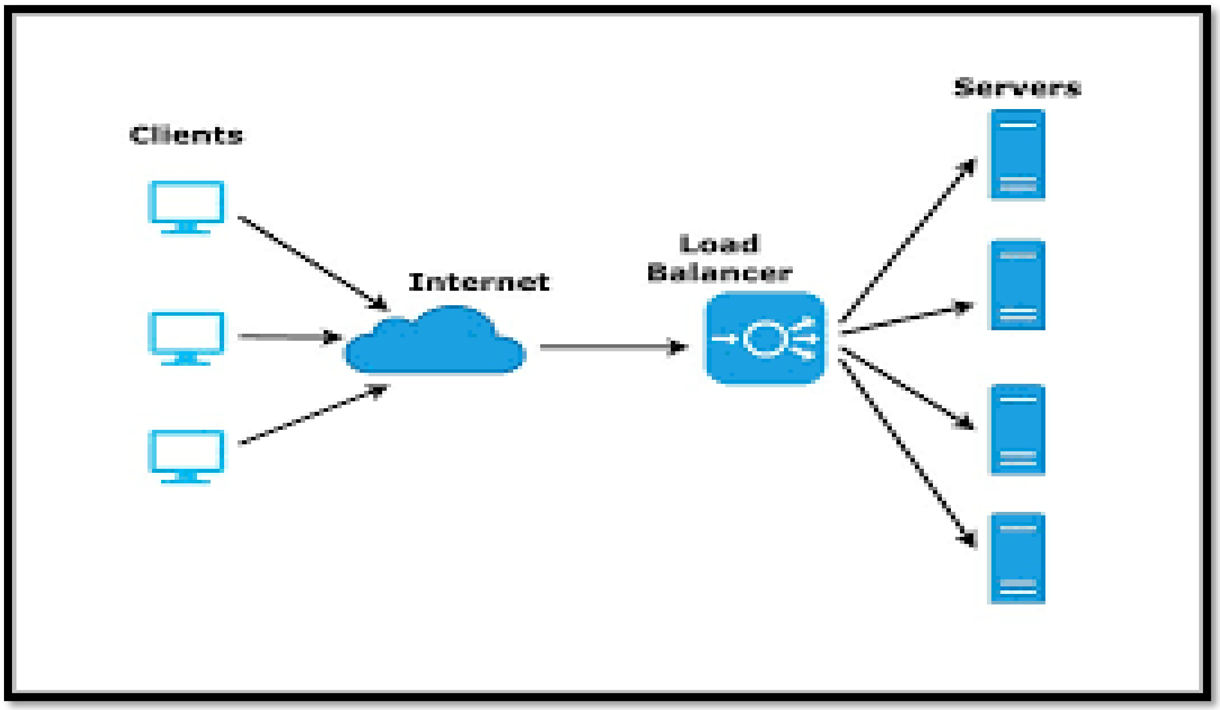
The k-means method has certain drawbacks.

- Choosing k by hand
- Reliance on beginning points.
- Different sizes and densities of data are grouped.
- Outliers are categorised and grouped together.
- Scaling with a large number of dimensions.



LOAD BALANCING

Load balancing is a technique for dispersing loads among a system's resources. As a result, load-balancing is critical in cloud-based architecture since each resource must perform the same amount of work at all times. The most crucial step is to provide certain techniques so that requests and solutions for each given request are balanced. In cloud load balancers, online traffic is automatically maintained by dispersing loads over various servers and resources. This has the benefit of increasing production while avoiding overload and shortening response time. The load balancing technique for CC application optimization is addressed in this work, and a summary is provided as an overview. Different algorithms are created for different purposes, for example, some algorithms are meant to achieve maximum throughput, while others are designed to have the shortest reaction time.



Different Efficient Algorithm techniques

The cloud deals with several things at once, from storing and retrieving of documents, sharing multimedia, fault tolerance, and allocating resources at a rapid rate. Therefore, it requires a proper load balancing to make it more efficient, responsive, reliable, and flexible.

Efficient task scheduling and resource management are challenging in distributed computing, but cloud engineers use genetic and conventional algorithms to enhance the performance of load balancing and to handle the operations intelligently. This post will look at five typical load balancing algorithms that improve scheduling, optimal resource allocation, etc.

1. Round Robin

Round Robin is one of the famous and commonly used load balancing algorithm, in which the processes are divided between processors. The process allocation order is kept locally independent from the remote processor allocations. In the round-robin, fixed quantum time is given to the job. The main emphasis in round-robin is on fairness and time limitation.

2. Weighted Round Robin (WRR)

Weighted Round Robin (WRR) scheduling facilitates controlled sharing of the network bandwidth. WRR assigns a weight to each queue; then, it is used to determine the amount of bandwidth allocated to the queue. The round-robin scheduling allows serving each queue in a set order, sending a limited amount of data before moving to the next queue and cycling back to the highest priority queue after servicing the lowest priority queue .

3. Least-Connections

One of the dynamic scheduling algorithms, the least-connection scheduling algorithm directs network connections to the server with the least number of established connections. To dynamically to estimate its load, it needs to count the number of connections for each server. The load balancer records each server’s connection number, increases a server’s connection number when a new connection is dispatched to it and decreases a server’s connection number when a connection is terminated or timeouts.

4. Weighted Least Connections

We have seen what Weighted Round Robin does to Round Robin. Weighted Least Connections algorithm does the same thing to Least Connections. It introduces a component of “weight,” based on each server’s respective capacities. As in the Weighted Round Robin, you will need to specify the “weight” of each server in advance.

5. Random

The random algorithm matches clients and servers by random, i.e., using a random number generator that underlies it. In cases where the load balancer receives a large number of requests, the requests will be distributed evenly to the nodes by a Random algorithm. Like Round Robin, the algorithm Random is sufficient for clusters that consist of nodes with similar configurations.

Now, we look at some of the bio-inspired dynamic load balancing algorithms, which are gaining popularity as load balancing techniques in cloud engineering. They mimic the natural behavior of living creatures, such as ants, bees, birds, and fishes, to improve the efficiency of other load balancing systems.

6. Ant Colony Algorithm

Ant colony algorithms apply the food searching behavior of ants in load balancing. Larger weight means that resource has a high power of computation. Load balancing ant colony optimization (LBACO) balances the load and minimizes make span. All tasks are assumed to be computationally intensive and independent of one another.

7. Honey Bee Foraging Algorithm

This algorithm is based on the foraging behavior of honey bees. When an underloaded VM assigns a task, it updates several priority tasks and the load of VM to other tasks on the waiting list. This approach aids other processes in selecting their VM. If a task has high priority, a VM with a minimum number of priority tasks is selected. It does not consider only load balancing but also keeps track of priorities of tasks which are currently removed from heavily loaded machines. It increases throughput and minimizes response time.

8. Throttled Load Balancing

This algorithm depends upon the theory of a suitable search for a virtual machine. The task manager makes a list of virtual machines. By using the list, the client request allotted to the relevant machine. If the machine's size and capability are suitable for request, th en the job is given to that machine. This algorithm is better than a round-robin algorithm.

9. Pareto Based Fruit Fly Optimization Algorithm

The Pareto-based fruit fly optimization algorithm (PFOA) is used to solve the task scheduling and resource allocating (TSRA) problem in a cloud computing environment. First, a heuristic based on the property of minimum cost initializes the population. Second, a resource reassign operator is used to generating non dominated solutions. Third, a critical path based search operator is used to improve exploitation capability.

10. Multi-Objective Scheduling Cuckoo Algorithm

CSA mimics the breeding behavior of cuckoos. Each individual searches for the most appropriate nest for the laying of an egg to maximize the survival rate of the egg and achieve the best habitat society. Fuzzy set theory is used to create the fuzzy search domain for membership, where it consists of all possible compromise solutions. CSA is searching for the best compromise solution within the fuzzy search d omain, tuning the fuzzy boundary design variables simultaneously. The tuning of fuzzy design variables eliminates the requirement of the expertise needed for setting these variables.

11. Min-Min Algorithm In A Cloud Environment

Load Balancing Min-Min algorithm has a three-level load balancing framework. Architecture at first level LBMM is the request manager who is responsible for receiving the task and assigning it to the service manager when the service manager receives the request. It divides it into subtasks and assigns the subtask to a service node based on node availability, remaining memory, and the rate of transmission that is responsible for performing the task.

LITERATURE SURVEY

This section consists of previous works that had been already proposed by several researchers. Some common approaches are also discussed here that work efficiently with response time, data center processing time and cost.

The work done by [4] proposed a novel load balancing algorithm called VectorDot. This algorithm handles the hierarchical complexity of the datacenter and multidimensionality of resource loads across servers network switches and storage in an agile data center that has integrated server and storage virtualization technologies.

The work done [5] proposed a mechanism CARTON for cloud control that unifies the use of LB and DRL. The LB (Load Balancing) is used to equally distribute the jobs to different servers so that the associated costs can be minimized and DRL (Distributed Rate Limiting) is used to make sure that the resources are distributed in a way to keep a fair resource allocation.

[6] addressed the problem of intra-cloud load balancing amongst physical hosts by adaptive live migration of virtual machines. The 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.

Work done by [7] presented an event driven load balancing algorithm for real-time Massively Multiplayer Online Games (MMOG).The algorithm after receiving capacity events as input, also analysis its components in context of the resources and the global state of the game session, then generating the game session load balancing actions.

The [8] proposed a scheduling strategy on load balancing of VM resources that uses historical data and current state of the system. Proposed strategy achieves the best load balancing and reduced dynamic migration by using a genetic algorithm.

The [9] proposed a Central Load Balancing Policy for Virtual Machines (CLBVM) that balances the load evenly in a distribute dvirtual machine/cloud computing environment.

The [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. The 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.

The [11] discussed a two-level task scheduling mechanism based on load balancing to meet dynamic requirements of users and obtain high resource utilization. Algorithm achieves load balancing by first mapping tasks to virtual machines and then virtual machines to host resources thereby improving the task response time, and resource utilization also overall performance of the cloud computing environment.

Author [12] investigated a decentralized honey bee based load balancing technique that is a nature inspired algorithm for selforganization. Algorithm 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. This is best suited for the conditions where the diverse population of service types is required.

RESOURCE SCHEDULING IN LOAD BALANCING

The main goal of any resource scheduling is to identify certain essential resources for an activity and determine the activity's start and end times depending on the resource's availability. When cloud resources are limited and the desire for additional work capacity is the major necessity, resource scheduling must be a top priority.

LOAD BALANCING FOR VIRTUAL MACHINES IN A CLOUD NETWORK

In order to achieve load balancing in cloud computing, numerous strategies for resource scheduling have been proposed. These methods are focused on virtual machine load balancing, which entails placing virtual machines on servers or hosts and balancing them. These algorithms are primarily targeted at increasing customer happiness while maximising resource usage to ensure that no one server is overwhelmed, hence facilitating the whole system's performance and execution.

The following are the elements that influence VM migration from one host to another:

1. A host's communication costs

2. Install on a server

3. A host's concurrent access

4. The time it takes for a programme to run on a host

5. A host's response time

6. A host's software and hardware limits

7. Moving a virtual machine (VM) from one cloud network to another. Homogeneous or heterogeneous cloud networks are possible.

The two methods for implementing the VM migration scheduling approach are as follows:

1. Static scheduling

2. Dynamic scheduling

CHALLENGES AND ISSUES IN LOAD BALANCING SCHEMES IN CLOUD COMPUTING

1. Optimal selection reduces the time it takes to migrate a project to a new computer.

2. Cost of communication inside the server or with servers outside the server

3. Workload distribution in a heterogeneous context with a focus on available resources

4. The most efficient use of resources within the limits

5. Maximum performance and reaction time during high load hours 6. Workflow robustness and consistency

7. Techniques for spotting errors must be accessible.

Tools and Technology used:

Hardware Requirements:

- RAM: 4GB and Higher
- Processor: Intel i3 and above
- Hard Disk: 500GB

Software Requirements:

 OS: Windows or Linux

 Python IDE: python 2.7.x and above

 Pycharm IDE required

- Jupyter notebook

 Setup tools and pip to be installed for 3.6 and above

 Language: Python

- Cloud Computing
- Machine learning (Unsupervised Learning)

FUNCTIONALITY/WORKING OF PROJECT:

The basic load balancing transaction is as follows:

1. The client attempts to connect with the service on the load balancer.

2. The load balancer accepts the connection, and after deciding 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

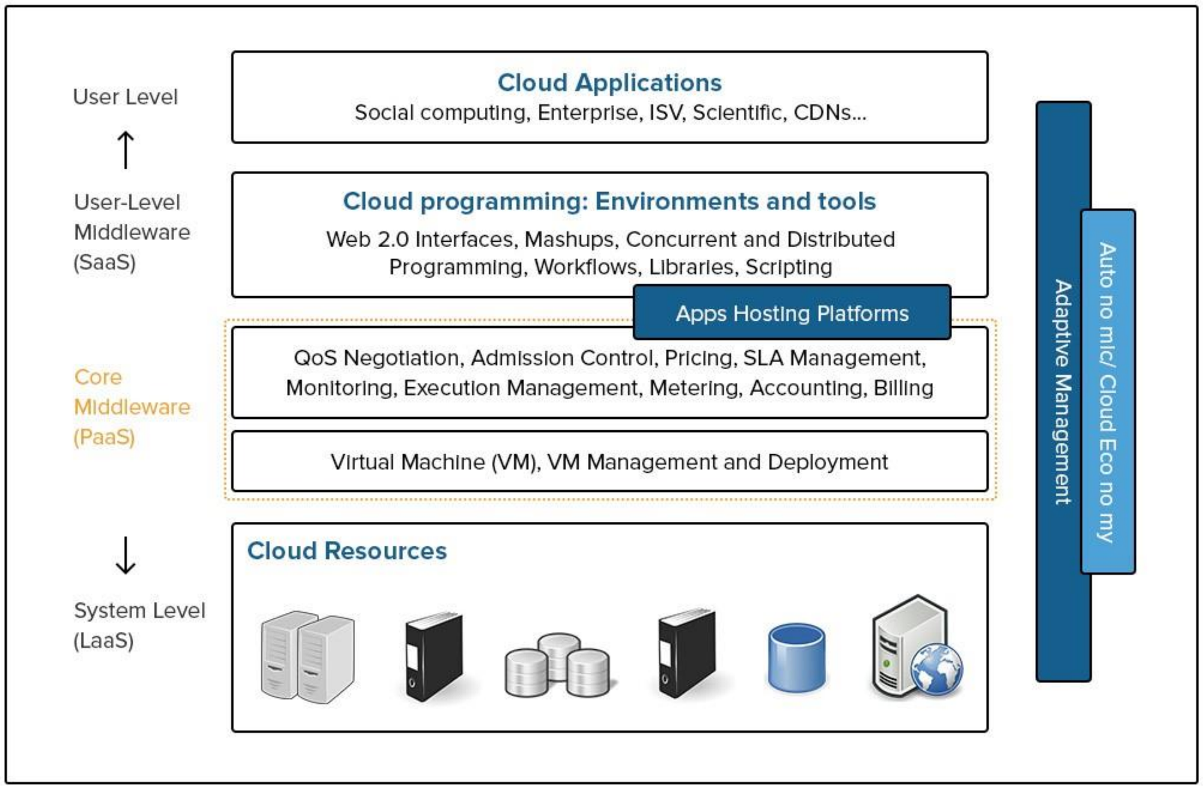
forwards the packet back to the client.

5. The client receives the return packet, believing that it came from the virtual

server, and continues the process.

This very simple example is relatively straightforward, 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.

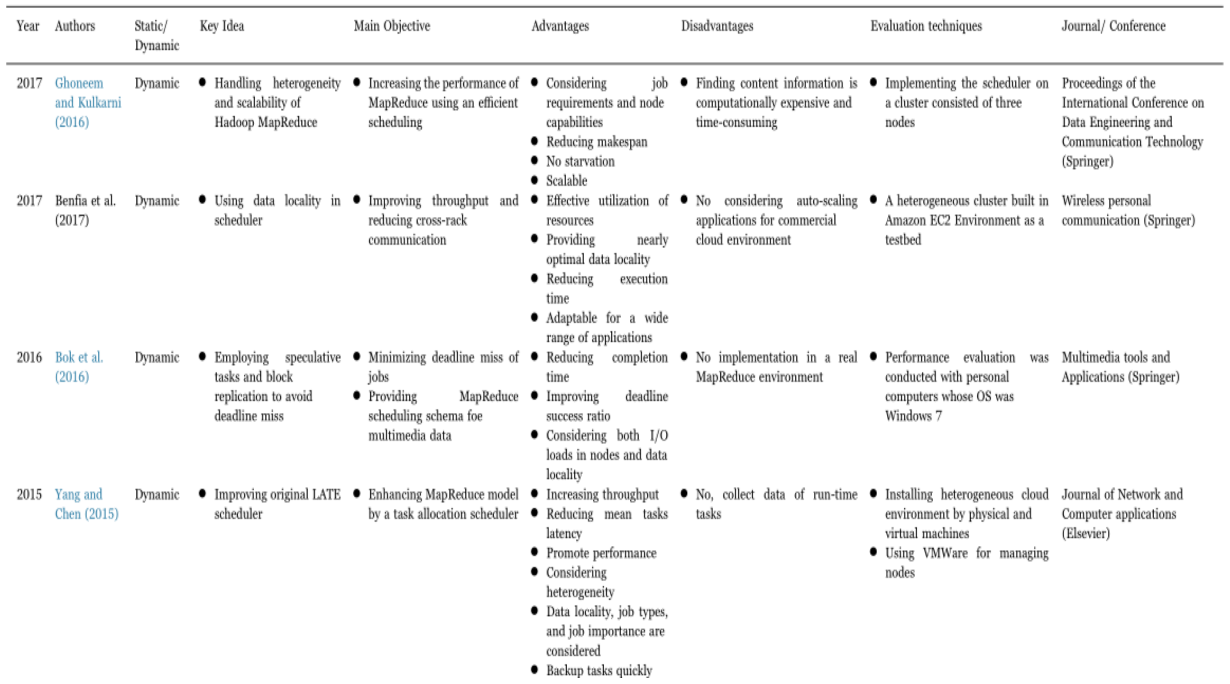
The Load Balancing Decision Usually at this point, two questions arise: how does the load balancer decide which host to send the connection to? And what happens if the selected host isn’t working? Let’s discuss the second question first. What happens if the selected host isn’t working? The simple answer is that it doesn’t respond to the client request and the connection attempt eventually times out and fails. This is obviously not a preferred circumstance, as it doesn’t ensure high availability. That’s why most load balancing technology includes some level of health monitoring that determines whether a host is actually available before attempting to send connections to it.


User Level
User-Level Middleware (SaaS)
Core Middleware (PaaS)
System Level (LaaS)
Cloud Applications
Social computing, Enterprise, ISV, Scientific, CDNs...
Cloud programming: Environments and tools
Web 2.0 Interfaces, Mashups, Concurrent and Distributed Programming, Workflows, Libraries, Scripting
Apps Hosting Platforms
QoS Negotiation, Admission Control, Pricing, SLA Management, Monitoring, Execution Management, Metering, Accounting, Billing
Virtual Machine (VM), VM Management and Deployment
Cloud Resources
Adaptive Management
Auto no mic/ Cloud Eco no my

Application Based Load balancing techniques

Prediction of multiple application services

There are multiple levels of health monitoring, each with increasing granularity and focus. A basic monitor would simply PING the host itself. If the host does not respond to PING, it is a good assumption that any services defined on the host are probably down and should be removed from the cluster of available services. Unfortunately, even if the host responds to PING, it doesn't necessarily mean the service itself is working. Therefore, most devices can do "service PINGs" of some kind, ranging from simple TCP connections all the way to interacting with the application via a scripted or intelligent interaction. These higher-level health monitors not only provide greater confidence in the availability of the actual services (as opposed to the host), but they also allow the load balancer to differentiate between multiple services on a single host. The load balancer understands that while one service might be unavailable, other services on the same host might be working just fine and should still be considered as valid destinations for user traffic. This brings us back to the first question: How does the load balancer decide which host to send a connection request to? Each virtual server has a specific dedicated cluster of services (listing the hosts that offer that service) which makes up the list of possibilities.

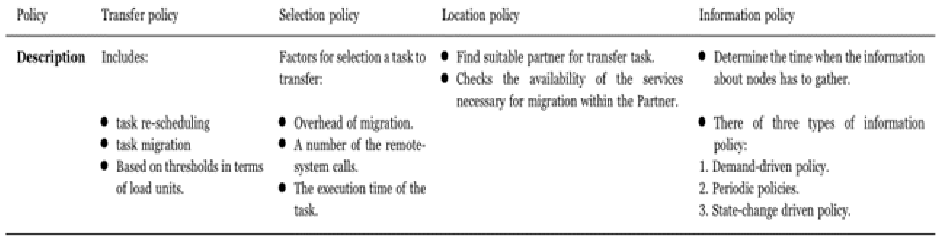


Year	Authors	Static/ Dynamic	Key Idea	Main Objective	Advantages	Disadvantages	Evaluation techniques	Journal/ Conference
2017	Ghoneem and Kulkarni (2016)	Dynamic	• Handling heterogeneity and scalability of Hadoop MapReduce	• Increasing the performance of MapReduce using an efficient scheduling	• Considering job requirements and node capabilities • Reducing makespan • No starvation • Scalable	• Finding content information is computationally expensive and time-consuming	• Implementing the scheduler on a cluster consisted of three nodes	Proceedings of the International Conference on Data Engineering and Communication Technology (Springer)
2017	Benfia et al. (2017)	Dynamic	• Using data locality in scheduler	• Improving throughput and reducing cross-rack communication	• Effective utilization of resources • Providing nearly optimal data locality • Reducing execution time • Adaptable for a wide range of applications	• No considering auto-scaling applications for commercial cloud environment	• A heterogeneous cluster built in Amazon EC2 Environment as a testbed	Wireless personal communication (Springer)
2016	Bok et al. (2016)	Dynamic	• Employing speculative tasks and block replication to avoid deadline miss	• Minimizing deadline miss of jobs • Providing MapReduce scheduling schema foe multimedia data	• Reducing completion time • Improving deadline success ratio • Considering both I/O loads in nodes and data locality	• No implementation in a real MapReduce environment	• Performance evaluation was conducted with personal computers whose OS was Windows 7	Multimedia tools and Applications (Springer)
2015	Yang and Chen (2015)	Dynamic	• Improving original LATE scheduler	• Enhancing MapReduce model by a task allocation scheduler	• Increasing throughput • Reducing mean tasks latency • Promote performance • Considering heterogeneity • Data locality, job types, and job importance are considered • Backup tasks quickly	• No, collect data of run-time tasks	• Installing heterogeneous cloud environment by physical and virtual machines • Using VMWare for managing nodes	Journal of Network and Computer applications (Elsevier)

Additionally, the health monitoring modifies that list to make a list of "currently available" hosts that provide the indicated service. It is this modified list from which the load balancer chooses the host that will receive a new connection. Deciding the exact host depends on the load balancing algorithm associated with that particular cluster. The most common is simple round-robin where the load balancer simply goes down the list starting at the top and allocates each new connection to the next host; when it reaches the bottom of the list, it simply starts again at the top. While this is simple and very predictable, it assumes that all connections will have a similar load and duration on the back-end host, which is not always true. More advanced algorithms use things like current-connection counts, host utilization, and even real-world response times for existing traffic to the host in order to pick the most appropriate host from the available cluster services. Sufficiently advanced load balancing systems will also be able to synthesize health monitoring information with load balancing algorithms to include an understanding of service dependency. This is the case when a single host has multiple services, all of which are necessary to complete the user's request. A common example would be in e-commerce situations where a single host will provide both standard HTTP services (port 80) as well as HTTPS (SSL/TLS at port 443). In many of these circumstances, you don't want a user going to a host that has one service operational, but not the other. In other words, if the HTTPS services should fail on a host, you also want that host's HTTP service to be taken out of the cluster list of available services. This functionality is increasingly important as HTTP-like services become more differentiated with XML and scripting.

Connection maintenances available in-service host

To Load Balance or Not to Load Balance? Load balancing in regards to picking an available service when a client initiates a transaction request is only half of the solution. Once the connection is established, the load balancer must keep track of whether the following traffic from that user should be load balanced. There are generally two specific issues with handling follow-on traffic once it has been load balanced: connection maintenance and persistence. Connection maintenance If the user is trying to utilize a long-lived TCP connection (telnet, FTP, and more) that doesn't immediately close, the load balancer must ensure that multiple data packets carried across that connection do not get load balanced to other available service hosts. This is connection maintenance and requ ires two key capabilities: 1) the ability to keep track of open connections and the host service they belong to; and 2) the ability to continue to monitor that connection so the connection table can be updated when the connection closes. This is rather sta ndard fare for most load balancers. Persistence Increasingly more common, however, is when the client uses multiple short-lived TCP connections (for example, HTTP) to accomplish a single task. In some cases, like standard web browsing, it doesn't matter and each new request can go to any of the back-end service hosts; however, there are many more instances (XML, e-commerce "shopping cart," HTTPS, and so on) where it is extremely important that multiple connections from the same user go to the same back-end service host and not be load balanced. This concept is called persistence, or server affinity. There are multiple ways to address this depending on the protocol and the desired results. For example, in modern HTTP transactions, the server can specify a “keep-alive” connection, which turns those multiple short-lived connections into a single long-lived connection that can be handled just like the other long-lived connections. However, this provides little relief. Even worse, as the use of web services increases, keeping all of these connections open longer than necessary would strain the resources of the entire system. In these cases, most load balancers provide other mechanisms for creating artificial server affinity. One of the most basic forms of persistence is source-address affinity.



Policy	Transfer policy	Selection policy	Location policy	Information policy
Description	Includes: • task re-scheduling • task migration • Based on thresholds in terms of load units.	Factors for selection a task to transfer: • Overhead of migration. • A number of the remote-system calls. • The execution time of the task.	• Find suitable partner for transfer task. • Checks the availability of the services necessary for migration within the Partner.	• Determine the time when the information about nodes has to gather. • There of three types of information policy: 1. Demand-driven policy. 2. Periodic policies. 3. State-change driven policy.

This involves simply recording the source IP address of incoming requests and the service host. they were load balanced to, and making all future transaction go to the same host. This is also an easy way to deal with application dependency as it can be applied across all virtual servers and all services. In practice however, the wide-spread use of proxy servers on the Internet and internally in enterprise networks renders this form of persistence almost useless; in theory it works, but proxy-servers inherently hide many users behind a single IP address resulting in none of those users being load balanced after the first user’s request—essentially nullifying the load balancing capability. Today, the intelligence of load balancer–based devices allows organizations to actually open up the data packets and create persistence tables for virtually anything within it. This enables them to use much more unique and identifiable information, such as user name, to maintain persistence. However, organizations one must take care to ensure that this identifiable client information will be present in every request made, as any packets without it will not be persisted and will be load balanced again, most likely breaking the application.

<u>CLOUD COMPUTING WORKFLOW</u>

The following is a summary of cloud computing workflow:

1. GET STARTED

2. Determine if the user should be granted access or not.

3. If no, display a notification stating that you are not permitted to use the cloud.

4. Add the person to the user queue if yes.

5. Does the Datacenter have sufficient resources to generate a virtual machine for the user?

6. If no, then wait for resources occupied by other operating users to be released.

7. If yes, construct a virtual machine (VM) for the user on a datacenter server.

8. Is a VM user sending Cloudlets?

9. If no, then don't do anything.

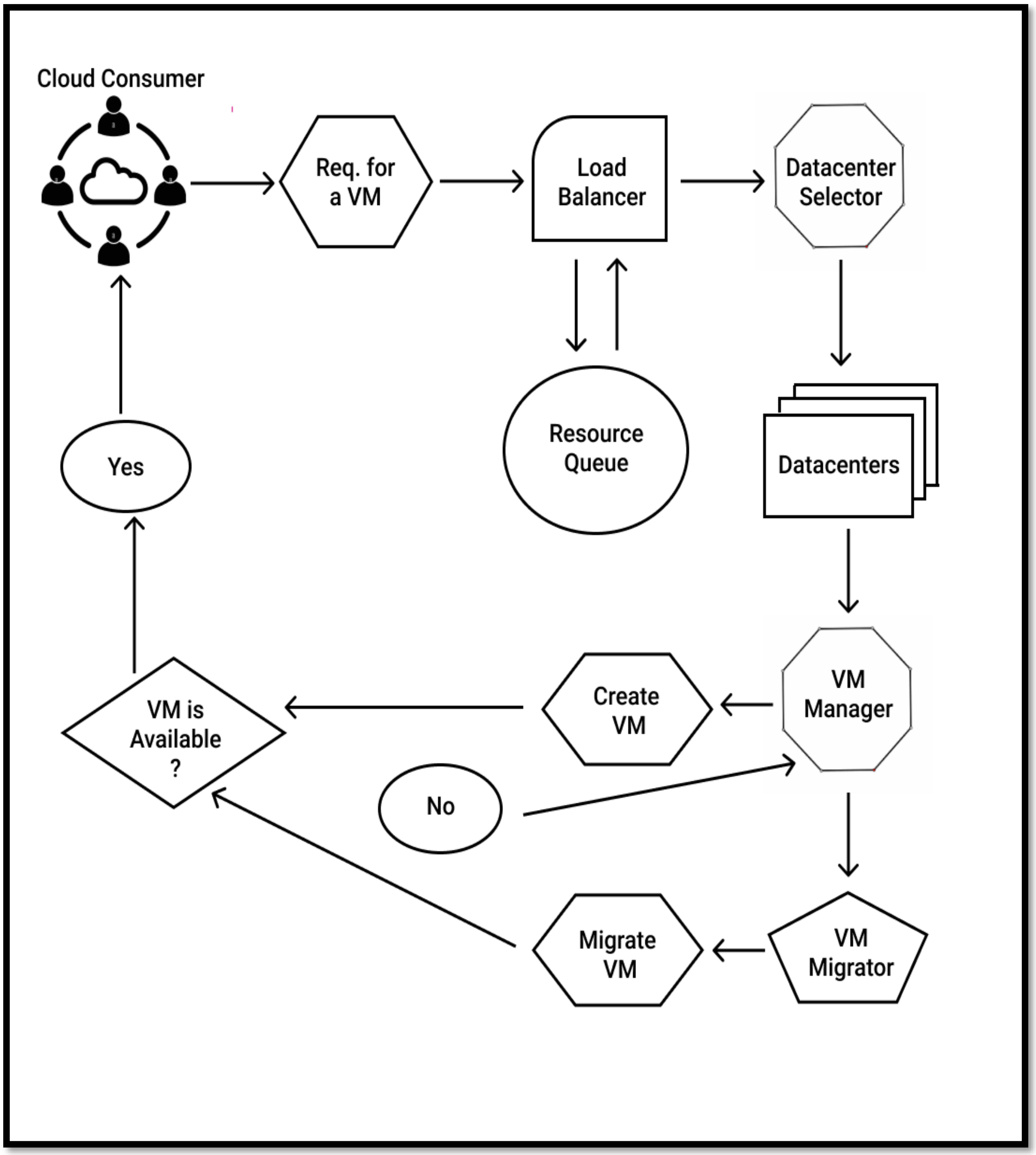
If so, then examine the hosting server's processing capabilities.

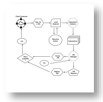
11. Submit cloudlets for processing if the hosting server has enough processing capability.

12. If no, then wait for a server to become available.

13. END

WORK FLOW DIAGRAM





CONCLUSION:

This separation of public air into a number of sub-clouds is completed in a good geographical location. This study elucidated the concept of cloud load balancing in a profitable way. It also included a thorough examination of the present and most widely used cloud load balancing strategies. The authors of this publication believe that their research will be useful to other researchers in the future.

WORK IN THE FUTURE

In the future, we will test the algorithm's usefulness and influence on load balancing, as well as apply it to a real-world scenario to improve outcomes and performance.

REFERENCES:

[1] John Harauz, Lorti M. Kaufinan. Bruce Potter, "Data Security in the World of Cloud Computing", IEEE Security & Privacy,

Co published by the IEEE Computer and Reliability Societies, July/August 2009.

[2] National Institute of Standards and Technology- Computer Security Resource Center -www.csrc.nist.gov

[3] http://en.wikipedia.org/wiki/Cloud_computing.

[4] Yashpalsinh Jadeja and Kirit Modi, “Cloud Computing - Concepts, Architecture and Challenges”, International Conference on

Computing, Electronics and Electrical Technologies [ICCEET], IEEE-2012.

[5] Ram Prassd Pandhy (107CS046), P Goutam Prasad rao (107CS039). “Load balancing in cloud computing system” Department

of computer science and engineering National Institute of Technology Rourkela, Rourkela-769008, Orissa, India May-2011.

[6] J. Sahoo, S. Mohapatra and R. lath "Virtualization: A survey on concepts, taxonomy and associated security issues" computer

and network technology (ICCNT), IEEE, pp. 222-226. April 2010.

[7] Bhaskar. R, Deepu.S. R and Dr.B. S. Shylaja "Dynamic Allocation Method For Efficient Load Balancing In Virtual Machines

For Cloud Computing Environment" September 2012.

[8] R.Shimonski. Windows 2000 & Windows server 2003 clustering and load balancing. Emeryville. McGraw-Hill Professional

publishing, CA, USA (2003), p 2, 2003.

[9] J. Kruskall and M. Liberman."The Symmetric Time Warping Problem: From Continuous to Discrete. In Time Warps, String

Edits and Macromolecules: The Theory and Practice of Sequence Comparison", pp. 125-161, Addison-Wesley Publishing Co.,

1983.

[10] Mr. Nitin S. More, Mrs. Swapnaja R. Hiray and Mrs. Smita Shukla Patel," Load Balancing and Resource Monitoring in

Cloud", International Journal of Advances in Computing and Information Researches ISSN: 22774068, Volume 1– No.2, April

2012.

[11] R. X. T. and X. F. Z,"A Load Balancing Strategy Based on the Combination of Static and Dynamic, in Database Technology

and Applications (DBTA)",2nd International Workshop,2010.

[12] M Randles, D. Lamb, and A. Taleb-Bendiab, "A comparative study into distributed load balancing algorithms for cloud

computing,” 2010 IEEE 24th international conference on advanced information networking and application workshops,2010, pp.

551-556.

Kolb, L., Thor, A., Rahm, E., 2011. Block-based Load Balancing for Entity Resolution with MapReduce. International Conference on Information and Knowledge Management (CIKM), 2397–2400.

Kolb, L., Thor, A., Rahm, E., 2012. Load Balancing for MapReduce-based Entity Resolution, IEEE In: Proceedings of the 28th International Conference on Data Engineering, 618-629.

Komarasamy, D., Muthuswamy, V., 2016. A novel approach for dynamic load balancing with effective Bin packing and VM reconfiguration in cloud. Indian J. Sci. Technol. 9 (11), 1–6.

Koomey, J.G., 2008. Worldwide electricity used in datacenters. Environ. Res. Lett. 3 (3), 034008.

Kulkarni, A.K., B, A, 2015. Load-balancing strategy for Optimal Peak Hour Performance in Cloud Datacenters. In: Proceedings of theIEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES).

Kumar, S., Rana, D.H., 2015. Various dynamic load-balancing algorithms in cloud environment: a survey. Int. J. Comput. Appl. 129 (6).

Lee, K.H., Choi, H., Moon, B., 2011. Parallel data processing with MapReduce: a survey. SIGMOD Rec. 40 (4), 11–20.

Li, R., Hu, H., Li, H., Wu, Y., Yang, J., 2015. MapReduce parallel programming model: a state-of-the-art survey. Int. J. Parallel Program., 1–35.

Lin, C.Y., Lin, Y.C., 2015. A Load-Balancing Algorithm for Hadoop Distributed File System, International Conference on Network-Based Information Systems.

Lua, Y., Xie, Q., Klito, G., Geller, A., Larus, J.R., Greenberg, A., 2011. Join-Idle-Queue: a

novel load-balancing algorithm for dynamically scalable web services. Int. J. Perform. Eval. 68, 1056–1071.

Malladi, R.R., 2015. An approach to load balancing In cloud computing. Int. J. Innov. Res. Sci. Eng. Technol. 4 (5), 3769–3777.

Manjaly, J.S., A, CE, 2013. Relative study on task schedulers in Hadoop MapReduce. Int. J. Adv. Res. Comput. Sci. Softw. Eng. 3 (5).

Shen, H., Sarker, A., Yuy, L., Feng Deng, F., 2016. Probabilistic Network-Aware Task Placement for MapReduce Scheduling. In: Proceedings of the IEEE International Conference on Cluster Computing.

Shen, H., Yu, L., Chen,L., Li, Z., 2016. Goodbye to Fixed Bandwidth Reservation: Job Scheduling with Elastic Bandwidth Reservation in Clouds. In: Proceedings of the International Conference on Cloud Computing Technology and Science.

Sidhu, A.K., Kinger, S., 2013. Analysis of load balancing techniques in cloud computing. Int. J. Comput. Technol. 4 (2).

Sim, K.M., 2011. Agent-based cloud computing. IEEE Trans. Serv. Comput. 5 (4), 564–577.

Singh, P., Baaga, P., Gupta, S., 2016. Assorted load-balancing algorithms in cloud computing: a survey". Int. J. Comput. Appl. 143 (7).

Singha, A., Juneja, D., Malhotra, M., 2015. Autonomous Agent Based Load-balancing algorithm in Cloud Computing. International Conference on Advanced Computing Technologies and Applications (ICACTA), 45, 832–841.

Sui, Z., Pallickara, S., 2011. A survey of load balancing techniques forData intensive computing. In. In: Furht, Borko, Escalante, Armando (Eds.), Handbook of Data Intensive Computing. Springer, New York, 157–168.

Tasquier, L., 2015. Agent based load-balancer for multi-cloud environments. Columbia Int. Publ. J. Cloud Comput. Res. 1 (1), 35–49.

Yang, S.J., Chen, Y.R., 2015. Design adaptive task allocation scheduler to improve MapReduce performance in heterogeneous clouds. J. Netw. Comput. Appl. 57, 61–70.

Zaharia, M., 2009. Job Scheduling with the Fair and Capacity Schedulers 9. Berkley University.

Zaharia, M., Borthakur, D., Sarma, J.S., 2010. Delay Scheduling: A Simple Technique for Achieving Locality and Fairness in Cluster Scheduling, in Proceedings of the European conference on Computer systems (EuroSys'10), 265–278.

Zaharia, M., Konwinski, A., Joseph, A.D., Katz, R., Stoica, I., 2008. Improving MapReduce Performance in Heterogeneous Environments. In: Proceedings of the 8th conference on Symposium on Opearting Systems Design and Implementation, 29–42.

Zhang, Y., Li, Y., 2015. An improved Adaptive workflow scheduling Algorithm in cloud environments. In: Proceedings of the Third International Conference on Advanced Cloud and Big Data, 112-116.

Dagli, M.K., Mehta, B.B., 2014. Big data and Hadoop: a review. Int. J. Appl. Res. Eng. Sci. 2 (2), 192.

Daraghmi, E.Y., Yuan, S.M., 2015. A small world based overlay network for improving dynamic load-balancing. J. Syst. Softw. 107, 187–203.

Dasgupta, K., Mandalb, B., Duttac, P., Mondald, J.K., Dame, S., 2013. A Genetic Algorithm (GA) based Load-balancing strategy for Cloud Computing, International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA), 10, 340-347.

Destanoğlu, O., Sevilgen, F.E., 2008. Randomized Hydrodynamic Load Balancing Approach, IEEE International Conference on Parallel Processing, 1, 196-203.

Deye, M.M., Slimani, Y., sene, M., 2013. Load Balancing approach for QoS management of multi-instance applications in Clouds. Proceeding on International Conference on

Cloud Computing and Big Data, 119–126.

Domanal, S.G., Reddy, G.R.M., 2015. Load Balancing in Cloud Environment using a Novel Hybrid Scheduling Algorithm. IEEE International Conference on Cloud Computing in Emerging Markets, 37-42.

Doulkeridis, C., Nørvåg, K., 2013. A survey of large-scale analytical query processing in MapReduce. VLDB J., 1–26.

Dsouza, M.B., 2015. A survey of Hadoop MapReduce scheduling algorithms. Int. J. Innov. Res. Comput. Commun. Eng. 3 (7).

Fadika, Z., Dede, E., Govidaraju, M., 2011. Benchmarking MapReduce Implementations for Application Usage Scenarios. In: 2011 IEEE/ACM Proceedings of the 12th International Conference on Grid Computing, 0, 90–97.