IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Improving Latency in Cloud Computing Using Master Node Consensus Approach

[1]Ossai Iyaye, M., [2]Nwiabu Nuka, D. [3]Taylor Onate, E.

[1]Mrs, [2]Dr, [3]Dr

[1]Department of Computer,

[1]Rivers State University, Port Harcourt, Nigeria

Abstract: Cloud computing is a technology that utilizes a group of interconnected nodes to provide a range of services such as computation, data processing, and resource sharing. The nodes work together to deliver these services within a data center, with a master node serving as the coordinator for the nodes in a cluster. This master node is responsible for managing tasks such as handling deadlocks, reducing latency, ensuring reliability, improving performance, and providing users with access to efficient processing services. Selecting the optimal master node can be a complex task, as it involves considering factors such as memory, CPU-MIPS, throughput, and bandwidth. The current methods for choosing master nodes are generally inflexible, as they do not consider the resources available on individual nodes. To address this issue, a "watcher" program is employed to continuously monitor the master node's fitness for coordinating the network of nodes. If the master node is no longer suitable, the watcher program triggers a real-time reassignment process, during which the next most suitable node is appointed as the new master. This process occurs quickly and seamlessly, so that the user is not aware of any delay in response time. The Master Node Consensus Approach (MNCA) method proposed in this paper aims to enhance efficiency, minimize latency, and improve overall user experience. To achieve this, the Master Node Consensus Approach (MNCA) utilizes a genetic algorithm (GA) to select the optimal master node from the data center. The proposed approach consists of three modules: initial populace creation, master node appointment, and candidate node appointment. In the initial module, a scheduling algorithm is used to randomly assign tasks to different nodes/servers, generating the initial population. The overall performance and capabilities of each node are then evaluated based on available resources. In the second module, the best optimal nodes are identified as potential master nodes using genetic operations like crossover, mutation, and fitness functions, with available resources being taken into consideration. From the set of optimal nodes, one node is selected to be the master node, while the others are designated as candidate nodes. In the third module, a candidate node is assigned as the new master node in the event of the original master node's failure.

Keywords **- Cloud computing, Latency, Consensus algorithm, Node, Master node**

I. INTRODUCTION

In the Greek allegory, creatures were taken off the surface of the Earth and placed in the night sky as constellations. Today, a similar phenomenon is occurring in the realm of computing. Programs and data are being transferred from desktop computers and corporate server rooms to the cloud[1]. Cloud computing was born with the introduction of the World Wide Web in 1990, and it is a disruptive invention when it comes to information technology. Although cloud computing eliminates the need for large capital investments in hardware and pricey information technology, it still has a long way to go[5].

In recent years, cloud computing has taken over many facets of our lives, including government, enterprise, industry, business, and markets. Cloud computing is a type of parallel computing and a distributed system that entails a collection of remote servers that are linked together to provide for the sharing of internet access to information technology, data processing, and centralized data storage[1].

Due to the massive amount of data generated, the demand for data storage is constantly growing, which propels the quick rise of the entire storage market. Since they offer data storage and management, cloud storage systems have developed into a fundamental part of the new era. Individual users, companies, and governments are all aggressively moving their data to the cloud right now. Such a large amount of data has the potential to be very profitable[9].

Vinothina[24] Said A breakthrough technology called "cloud computing" enables a third-party "cloud provider" to offer services to clients at any time, from any location, and under any circumstance. To provide clients with cloud resources and suit their demands, cloud computing uses virtualization and resource provisioning techniques. It is the process of making shared virtualized resources available to customers.

We are in the era of data. The steady rise in computing power over the past two decades has resulted in an enormous data flow. As a result, there appears to be a noticeable a discrepancy exists between the amount of data produced and the capacity of traditional systems to store, process, and effectively utilize this data. Because of its financial benefits, cloud computing has been increasingly popular in recent years. In particular, the deployment of data-intensive applications has been promised a variety of benefits from cloud computing's hosting[21].

Leading software companies like IBM, eBay, HP, Microsoft, Amazon, and Google have the chance to move their current operations into the cloud thanks to the flexible, elastic, and adaptable characteristics of cloud computing. With such a wide range of cloud service providers, it goes without saying that clients frequently have many options when selecting the finest service providers. The significant advancement and increase in cloud computing usage need for effective and precise cloud service selection procedures and accurate provider selection is crucial to raising the level of confidence between customers and providers[19].

A wide concept known as "cloud computing" provides customers with a range of infrastructure-based services, platform as a production tool, software as a service, and on-demand application licenses. These services are utilized by all types of clouds, including public, private, hybrid, and community clouds. The idea of "cloud computing" allows computers and other devices to exchange data, software, and resources over the internet. Currently, there are four key cloud service providers: Microsoft, IBM, Amazon, and Google. With the introduction of cloud computing, such massive amounts of data can be handled because of on-demand services[6]. According to Dimpi [8] a cloud offers its user a welcoming atmosphere as well as many services, including infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS).

Software as a service (SaaS) is the hosting of applications in the cloud and giving users online access to them. As a result, it can be said that cloud computing is used to offer enormous storage capacity on a remote server that is accessible from anywhere, and that cloud computing works in tandem with the big data concept, which is used to manage enormous amounts of data and extract meaningful information [13].

Infrastructure-as-a-Service (IaaS) is a specific kind of cloud computing service that provides basic computation, storage, and networking resources on demand and on a pay-as-you-go basis. An organization's infrastructure can be migrated to an IaaS service to save money on hardware expenditures, lower maintenance of on-premises data centers, and obtain real-time business analytics. IaaS solutions provide you the freedom to adjust the amount of IT resources you have according to demand. IaaS aid in enhancing infrastructure stability and speedy provisioning of new application and gives users a container environment in which to deploy their bespoke apps and dependencies, such as Google App Engine, Rackspace, Digital Ocean, Google Compute Engine etc.[10].

PaaS provides hardware platforms in the form of services such as database services and network services such as meta-heuristics for cloud task scheduling, followed by a thorough systematic evaluation that includes a new taxonomy of those techniques, as well as their benefits and drawbacks [27].

According to Satyanarayana [20], On-demand self-service, broad network access, resource pooling, rapid flexibility, and measured services are the five fundamental characteristics of cloud computing. Additionally, it has three deployment models. hybrid, public, and private. The advantages of employing the cloud to supply different processing, storage, and analytical services via the internet using hardware and software components cannot be overstated. Thanks to cloud computing, users can effortlessly access their files and use services like Gmail, Dropbox, and Skype from any device via the Internet. The cloud is a sizable collection of accessible and employable virtualized resources. Because of their low costs, cloud computing platforms have emerged as the most significant and well-known thing in the IT sector. Large companies like Microsoft, Amazon, and Google have adopted cloud computing as a result. The front end and the back end are the two components that make up the cloud computing system[23].

You could believe that something as fantastic as cloud computing won't present you with many challenges. But just like everything else, cloud computing has advantages and disadvantages. Adopting it presents several difficulties and hazards for businesses or individuals. Some of the challenges with cloud computing are data security and privacy, reduced visibility and control, improper access control and management, lack of expertise, cloud migration, latency[4].

Security and privacy:

Many consumers are unsure how secure their data is in the cloud and whether it is at risk. These reservations stem from a lack of guarantees in the event of a data loss or breach. These concerns could be addressed legally by Service Level Agreements (SLAs) that clearly establish legal promises regarding privacy and data security. We examine technological innovations and methods that could address this obstacle because security and privacy are significant issues in cloud technology [11]. An adequate assurance for the dual security of edge computing service users and service providers is the problem of mutual trust between users and edge nodes in an edge computing environment. Edge computing security faces a significant difficulty in figuring out how to achieve user security and credibility as well as guarantee the confidence of the edge service nodes[26].

Reduced visibility and control

Due to the absence of visibility on the local domain's networks, service function chaining placement encounters new difficulties in multi-domain environments. In fact, the domain owners are frequently reticent to disclose their topology to outsiders [16].

Latency

In general terms, latency refers to a delay in time between the cause and effect of a physical change being observed in a system. Within gaming communities, this delay is referred to as lag and is often caused by network delays in online games, resulting in a delay between the input to a simulation and the corresponding visual or auditory response. Latency [14] is a delay measurement that represents the time taken for data to travel from one point to another, such as from point A to point B on a screen. In a network, latency is measured as the time it takes for data to reach its destination, typically an ISP server, and then return in a round-trip delay. When it comes to streaming, latency refers to the delay or lag between the time a video is recorded or streamed in real life and when it is displayed on users' devices and reflected on screens. As data is passed from one source to another, it accumulates delay at each step of the streaming workflow. Latency is usually measured in milliseconds(ms)[2]. Some persons have shown different latency reduction mechanisms.

Approaches to Latency Reduction

Network bandwidth

Network congestion is a frequent source of slowness in cloud computing, which has grown substantially in recent years. Real-time cloud services, including video conferencing, online gaming, virtual desktops, and commercial transactions, need high availability and quick reaction times. Various alternative traffic route suggestions are used to avoid congestion. Due to the dispersed nature of the cloud and its diverse use patterns, it may be difficult to define cloud service reaction time or service latency. Nonetheless, some individuals have developed latency-reduction strategies that may improve the efficiency of cloud computing[1]. There are two popular latency measurement techniques, including:

a. **Round Trip Time (RTT):** This method involves measuring the time it takes for a request to be sent from a user to a server and for the corresponding response to be received. The round-trip time can be calculated by subtracting the time at which the request was sent from the time at which the response was received.

b. **Time to First Byte (TTFB):** This method involves measuring the amount of time it takes for a user to receive the first byte of data after sending a request. The TTFB can be calculated by subtracting the time at which the request was sent from the time at which the first byte of data was received.

Intelligent virtual machine placement

To reduce data access latency, it is essential to position virtual machines (VMs) based on the known location of data sets. Multiple VMs are used by cloud applications to process data sets, and completion time is a crucial performance[28] factor that is affected by several factors, including job scheduling, server load, data access bottlenecks, communication, and access latencies from processing nodes to data nodes. VMs must be positioned optimally to reduce the influence of data access latencies on completion times, particularly when it is not feasible to keep all data access local. The authors describe techniques for assigning VM locations that understand the obvious tradeoffs between lowering latencies and paying for bandwidth and that fulfill various restrictions. In addition, they investigate the problem of inter-VM latency constraints and give a practical heuristic for this circumstance[30]. The authors study the trade-off between different latency measurements and limitations using extensive simulations and typical linear assignment approaches for their algorithms. The location of virtual machines is crucial for improved performance and lower bandwidth costs. Lalith [12] adopted a virtualization strategy for optimizing thin client performance in cloud computing settings by reducing network latency. They conducted experiments with thin clients in various contexts in order to draw conclusions and suggest new strategies for lowering latency. Thin clients and virtual desktops are the ideal means to access cloud-based online services through the Internet. The research underlines the need of minimizing network latency for both real-time thin clients and cloud computing environments in order to enhance their performance and user experience. The report also proposes novel solutions to these issues.

Indeed, choosing the optimal leader node is a challenging task as it requires considering several factors, such as the node's processing power, network bandwidth etc. Moreover, the node's reliability and fault-tolerance capability are also essential factors to consider when selecting the leader node. For example, if a leader node fails, the entire network may become unstable, and data may be lost. Hence, it is crucial to choose a node that is reliable and can handle potential failures.
There are several techniques for selecting[29] the optimal leader node, including leader election algorithms, which enable the nodes to elect the leader among themselves. These algorithms ensure that the elected leader is the most suitable node based on various criteria, such as processing power, network bandwidth, and latency. Another technique is load balancing, which involves distributing the workload across all nodes in the network to reduce latency and improve performance.

Overall, selecting the optimal leader node is an important task in ensuring the stability and reliability of cloud computing platforms[3].

II. RELATED WORKS

In their 2016 paper, Ktari et al. evaluated methods for choosing a frequent leader or head node and offered a dynamic tree-based agent-based strategy. The goal was to maintain a forest structure in which the node with the highest ID would always be the leader. During the procedure for selecting a leader, the ID number was generated at random, without taking into account any particulars or resources.

The probabilistic analysis of traffic signals was used by Hanan [17] to provide a strategy for leader selection. Hanan and coworkers suggested a method for choosing a leader that took cues from the randomness of traffic signals. By treating the leader election process as a traffic light control issue, they were able to replicate it using the Markov chain model. The suggested approach was developed with the intention of enhancing the efficiency and reliability of leader election in distributed systems.
In Nassima[16] algorithm, all nodes are connected using a unidirectional interface, and a priority number is generated at random to determine the leader node. If the leader node fails, the process is repeated to select a new leader. This process involves communication of 2(n-1) messages across the network, with (n-1) messages sent initially and another (n-1) messages sent to choose the new leader.

Garcia-Molina proposed a leader selection algorithm that takes into account the requirement relations among the nodes in the system. The algorithm ensures that the selected leader satisfies all requirements, and that the requirements are consistent with each other. The algorithm works by first selecting a candidate leader based on some criteria (such as the node with the highest ID), and then checking if the candidate satisfies all requirements. If the candidate does not satisfy the requirements, another candidate is selected and the process is repeated. The algorithm terminates when a leader is found that satisfies all requirements.

In Jiawei [10] *et al.* (2020), the existing leader selection methods such as the traditional ring algorithm and the bully algorithm had limitations in terms of message complexity and leader nomination speed. To address these issues, an extended form of the bully algorithm was proposed for leader selection in cloud computing, where a Super Node (SN) was introduced to improve leader nomination speed and minimize message complexity. The TOSL approach, which had a leader election complexity of O(k2n2), was also mentioned as a less efficient method compared to the extended bully algorithm. Similarly, Rakesh[19] imply that if the leader fails, the network would too, and that will need a full restart of the process.

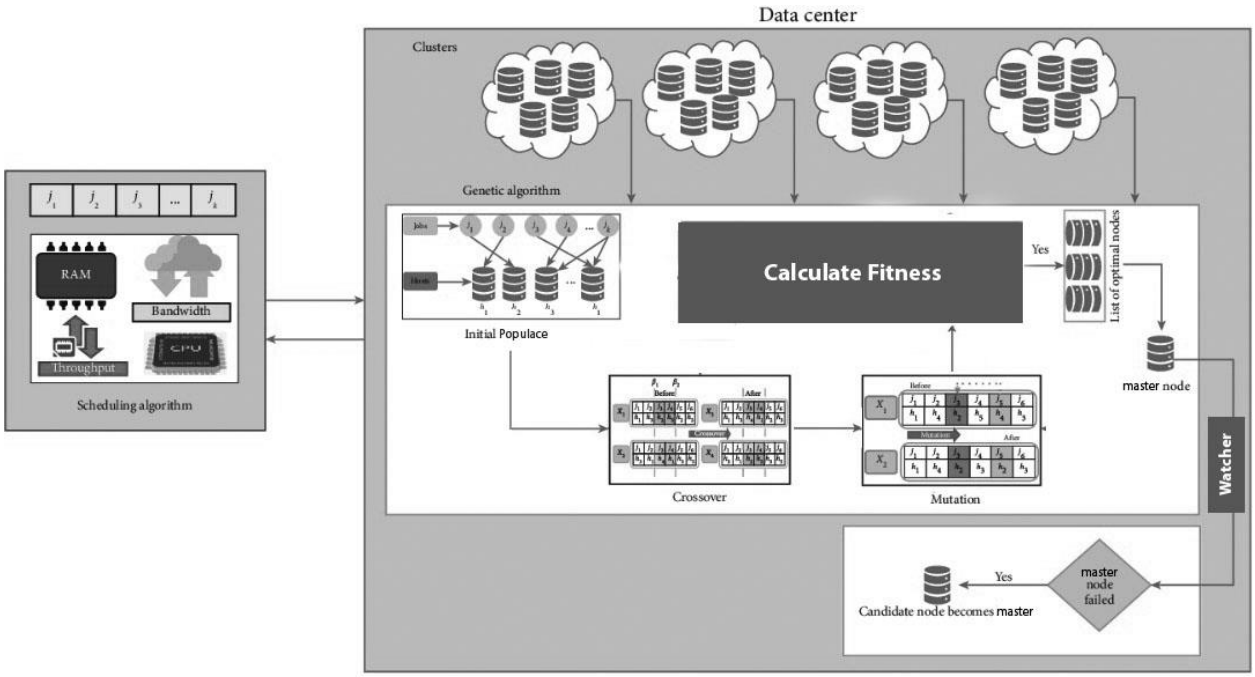
Satyanarayana [20] showed the election complexity of $O(logn)$ and the message complexity is $O(k2n2)$.

III. DESIGN

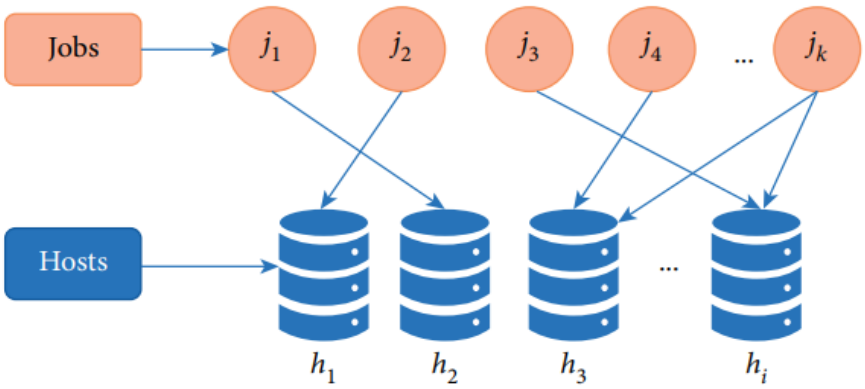
The Abstract Description of the MNCA Algorithm

The study is broken down into three sections. Three separate yet interdependent modules—"initial populace formation module," "master node appointment module," and "candidate node appointment module"—work together for peak network efficiency. Take into account a Data Center (DC) with multiple servers called hosts $H = \{h_1, h_2, h_3 \ldots, h_i\}$ that are grouped in the form of cluster $C = \{C_1, C_2, C_3 \ldots\ldots C_y\}$ and each host h_i has $R_m = \{R_{hi1}, R_{hi2}, R_{hi3}, \ldots\ldots, R_{him}\}$ resources, for example, bandwidth, CPU, throughput, and memory. In the data center, jobs like $J = \{j_o, j_1, j_2, \ldots\ldots, j_k\}$ that contain multiple tasks are submitted to each cluster C_y. These jobs or tasks are randomly mapped on different hosts H_i that can make chromosomes sets like

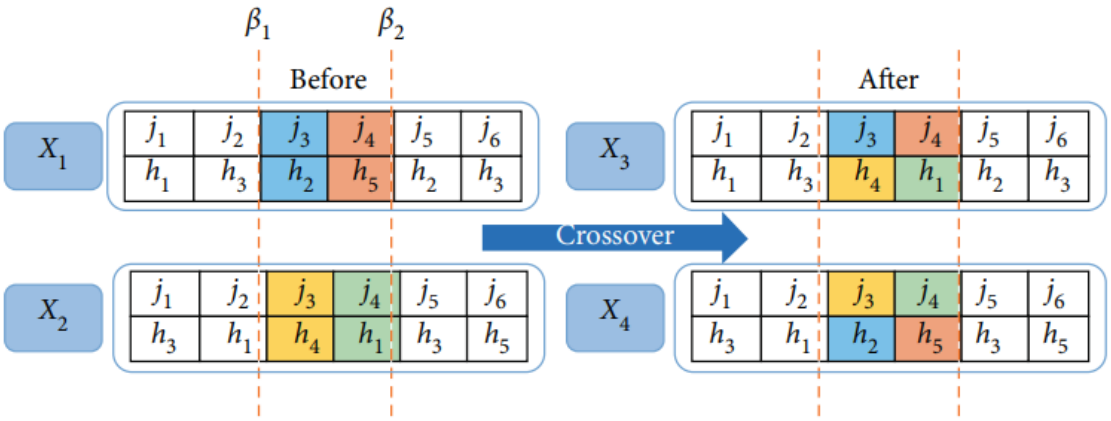
$X = \{x_1, x_2, x_3, \ldots\ldots, x_z\}$, where $X_1 = \{h_1 j_1, h_2 j_3, h_3 j_2, \ldots\ldots, h_i j_k\}$ and hi, jk_i represents genes in a chromosome. There are several numbers of chromosomes in each set of the population represented as $P = \{p_1, p_2\, p_3, \ldots\ldots, p_q\}$. Next, the algorithm performs crossover and mutation operations on the chromosomes and evaluates them using the fitness function. Afterward, the chromosomes are ranked based on their fitness, and the Master nodes are selected from the highest-ranked ones. The Master nodes are responsible for allocating jobs to each node in the cluster to improve the efficient load balancing of the network.



(a) The abstract view of the Master Node Consensus Approach (MNCA) Model



(b) The initial populace creation



(c) Crossover on Job and Host

Figure 1: High level description of Master Node Consensus Approach Algorithm

Initial Populace Formulation

The suggested system's initial population is produced when a scheduling algorithm is used to distribute work across many hosts. The suggested technique employs the following strategy for seed population when scheduling tasks between intermediate nodes is as follows : $j_1$on h_2, $j_2$on h_1, $j_3$on h_1, $j_4$on h_3,(there are several numbers of chromosomes in each set of the population represented as $P_q = X_1, X_2, X_3, \ldots\ldots\ldots, X_z$, each set of chromosomes contains multiple genes, and the genes represent map jobs on different hosts.

For example, the chromosome X_1 is represented as $X_1 = \{h_1j_2, h_2j_2, h_1j_3, \ldots\ldots, h_{(i-1)}j_{(k-1)}, h_ij_k\}$

Master Node Appointment

In this module, the algorithm executes some genetic operations, including crossover and mutation. Two chromosomes (isolated from the population at large) are chosen at random to undergo the crossover process. The suggested method employs a two-cut-point strategy for crossing, one of several possible approaches. The technique uses the following equation to choose two random places for the cuts:

$$X_1 = \{h_1j_2, h_2j_2, h_1j_3, \ldots\ldots, h_{(i-1)}j_{(k-1)}, h_ij_k\} \quad (1)$$

To select the two cut-point positions, β1 and β2, the algorithm generates a random number using the equation rand(2)(k1) + 1. These two chromosomal sets are selected because they have the greatest fitness values and are used to determine the cutting places. Crossover is conducted to produce two new individuals, or offspring, in the population after two random cut points, 1 and 2, are chosen, as seen in Figure 1c. This method improves the following generation of the population, and it is continued until the optimal answer is found.

Mutation

The program then carries out the mutation process on the progeny after the crossover procedure has finished. This procedure is unary since it affects just one trait in the offspring. During the mutation process, variations are minimal as a small value is typically chosen. In the suggested technique, a random location in the offspring is chosen, and the value there is swapped with another value, as in the replacement process for mutation as shown in Figure 3.5.

A fitness function, Fc, is used to measure the efficacy of the suggested procedure; it determines the optimal solution by solving the following equation:

$$F_c = max\sum_{i=0}^{n} f_h \quad (2)$$

where f_h represents a host's fitness as determined by the following formula:

$$f_h = \Theta_1 C_\alpha + \Theta_2 R_\alpha + \Theta_3 B_\alpha + \Theta_4 T_\alpha, \quad (3)$$

The fitness function (Fc) takes into account system parameters such as CPU-MIPS, memory, bandwidth, and throughput. These parameters are adjusted based on the Service Level Agreement (SLA) using the knapsack algorithm, as shown in Table 1. The weights of these parameters determine their impact on the fitness value, with higher weight values having a greater impact than lower weight values. The suggested technique employs a tournament to choose who will serve as the master node or candidate node. The winner is the host with the greatest fitness value, as determined by applying the fitness function to a population produced at random.

Algorithm for Master Node Consensus Approach

Appointment of Candidate Node

Each cluster will have a designated leader after GA has been applied. Each cluster C in a data center will have a candidate node selected to become the master node in case the master node fails within a certain window of time, the candidate node is selected to take over. One of the best hosts from each cluster is chosen to be the master and the other is put up as a candidate using the described method. Algorithm 1 is used by the MNCA system after each time interval to optimize the master and candidate nodes.

Input: List of Jobs and Host
Output: Master Node, Candidate Node

```
1.  populance = randomly create
2.  while i = 1 to total populance
3.     poplace list = randomly assign each job to a host
4.  end while;
5.  calculate the cost ();
6.  if time ( == scheduling interim)
7.     for (!Stop criteria)
8.       perform crossover();
9.       perform mutation();
10.      calculate fitness();
11.  end for
12.  master and candidate node = host with maximal value from (10)
        in the initial index of populace list is master node and the succeeding candidate
13.    if (master node failed)
14.    candidate node = host at the succeeding initial of populace will be the master node.
15.  endif
```

IV. RESULTS AND DISCUSSIONS

4.1 Performance Evaluation

To test the effectiveness of this approach, we increased the number of tasks while maintaining the same number of hosts (50 in total). From 200 to 20,000 tasks were available, always in 200-task increments. Figures 2 depicts the findings, which revealed that the suggested strategy worked well in terms of execution time. When compared to other algorithms like BLA and HEFT, the proposed MNCA algorithm has faster throughput. Experiment findings showed that compared to the BLA and HEFT algorithms, the MNCA resulted in faster task execution times.

4.2 Evaluation and Testing

The suggested method's efficacy and efficiency were evaluated via a series of tests using CloudSim plus simulator, with results compared to those of various modern algorithms. The following describes the outcomes of various trials.

Table 1: First Experiment: Performance Evaluation through Node/Host Increment

Nodes	MNCA	HEFT	BLA
20	5900	11000	7000
40	4000	7000	4500
60	3000	6000	4000
80	2000	4000	3000

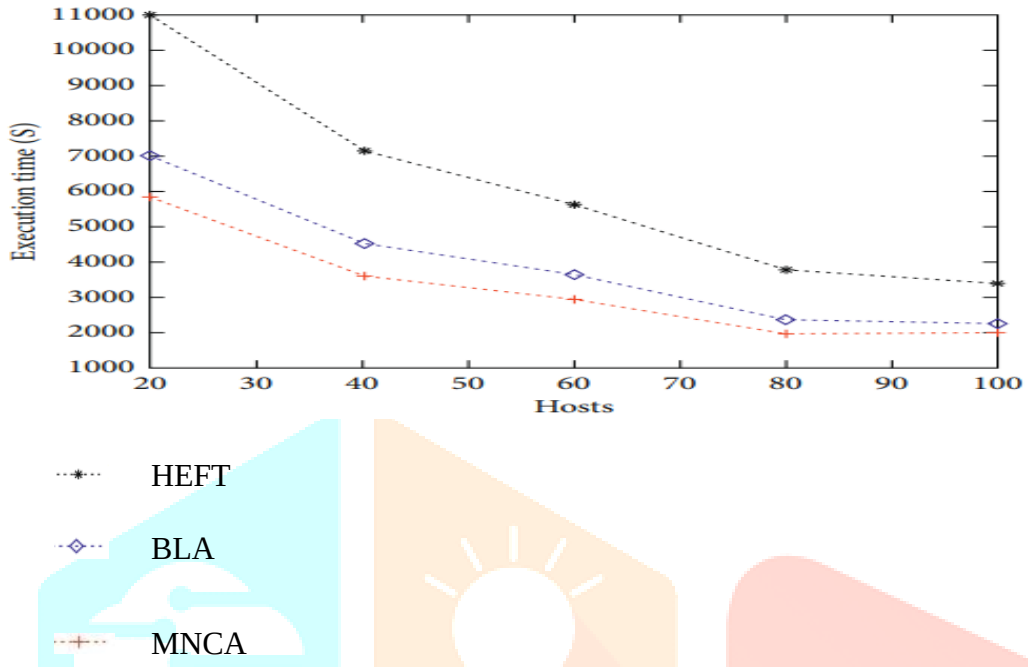


Figure 3: Memory Utilization

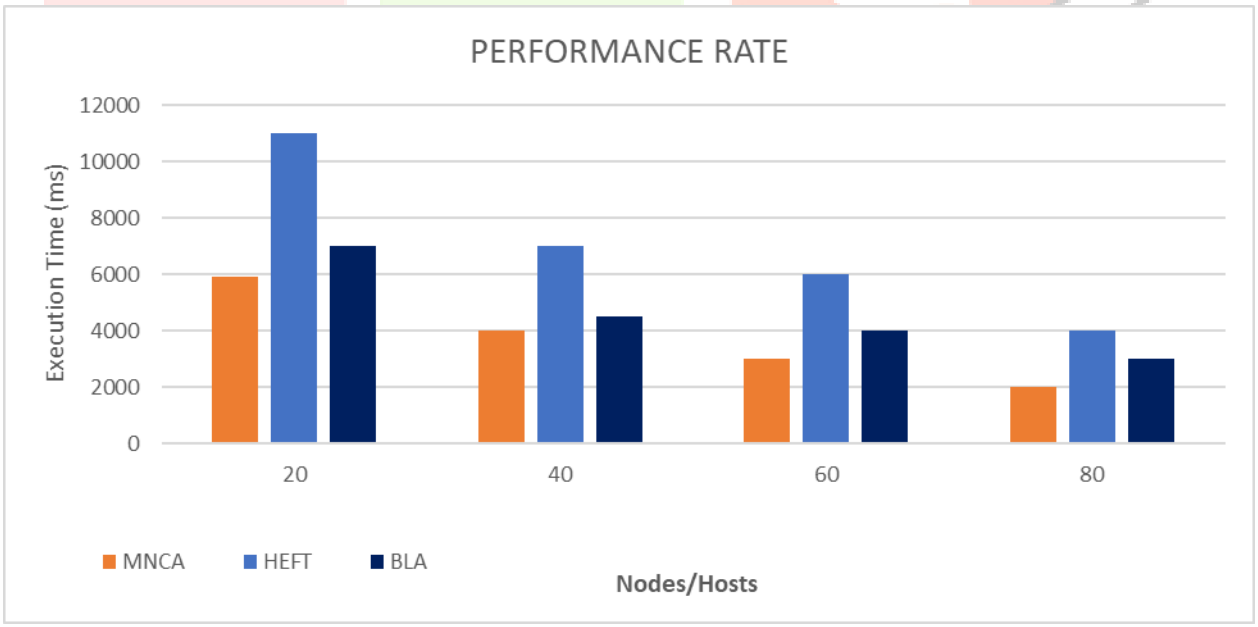


Figure 4: Performance rate when node is increased

Figure 4 validates that the performance of MNCA is better compared to BLA and HEFT algorithms in terms of execution time.

Using metrics including runtime, power use, memory utilization, and service level agreement (SLA) breach, the first experiment intended to gauge how well the suggested MNCA algorithm performed in comparison to various state-of-the-art algorithms. In this experiment, we scaled up the number of hosts/servers while maintaining the same 1000 jobs. The host range was established as 20–100 with a 20–increment. The

findings shows that the suggested MNCA algorithm achieved the fastest work completion time compared to the state-of-the-art algorithms BLA and HEFT. As can be seen in Figures 2, the algorithm also fared better in terms of memory use. When the number of hosts is increased while the number of jobs is kept constant, memory consumption improves. In conclusion, the first experiment proves that the suggested MNCA algorithm has superior performance and efficiency.

4.3 Evaluation through the Task/Job Increment

Several simulations were run in the CloudSim Plus environment to test the efficacy of the proposed approach. The trials compared the suggested technique to existing cutting-edge algorithms for both efficacy and efficiency. Execution time, energy use, memory utilization were among the metrics evaluated for analysis. In the first experiment, we discovered that the proposed MNCA outperformed state-of-the-art algorithms in terms of execution time, memory usage, energy consumption. The second study increased the number of tasks while maintaining the same number of hosts. In comparison to the BLA and HEFT algorithms, the suggested technique was shown to have faster execution times.

4.4 Evaluation through Different Weightages (Θ)

Using varied weights for criteria including CPU-MIPS, memory, bandwidth, and throughput dependent on user needs, this experiment compared MNCA's performance efficiency to that of BLA and HEFT algorithms. The parameter weights changed based on which of two scenarios were chosen. In the first example, the hosts remained constant, but the workload did not, leading to varying priorities among memory, throughput, bandwidth, and processing power. With respect to execution time, the MNCA was superior to other algorithms. The second scenario also made use of all parameters, but assigned different weights to them. The number of jobs and the number of nodes per host were both the same. When compared to the BLA and HEFT algorithms, the MNCA demonstrated superior efficiency. In the second trial, we raised the workload but kept the number of hosts same. In terms of turnaround time, the MNCA performed well. Based on execution time, all testing findings verify the MNCA's performance efficiency. In comparison to the BLA and HEFT algorithms, the suggested approach speeds up process execution.

V. Conclusion

The selection of a master node in a cloud data center is the primary emphasis of our suggested method. The primary function of the master node is to fulfill client needs by executing tasks and jobs effectively. Many methods have been proposed to deal with this problem, but many of them ignore the host/ server's available resources. To get around this problem, we used ideas from the genetic algorithm to create a new algorithm we dubbed MNCA. The master and candidate nodes in MNCA are chosen at random from among all accessible nodes. This makes sure the candidate node takes over immediately if the master node fails within a certain time frame. We ran simulations comparing MNCA's performance to that of state-of-the-art algorithms, and found that MNCA was superior in all metrics including runtime, SLA observance, memory use, and power consumption. We want to build additional protocols and use this method in IoT settings, wireless networks, and fog computing in the near future.

5.1 Recommendation

The findings of this paper are recommended to cloud administrators and data centre environment, software developers and researchers with keen interest in the study area. This is because task communication and synchronization amongst nodes is very important f0r preventing network from becoming unstable. To avoid network instability, it is crucial that a single node be chosen as the network's leader.

5.1.1 Future Work

Current and future work is focused on implementing this algorithm to enhance master node election in a distributed system within the 5G/6G networking environment for cloud computing.

REFERENCES

[1] Adh, R., D, R. & Shahidul, I. (2020). Fog-based Spider Web Algorithm to Overcome Latency in Cloud Computing. *Iraqi Journal of Service*, 1781-1790.

[2] Alemnew, A., Steffie, E., Vaibhav, B., Pasi, S., Jörg, O. & Asrese, E. (2019). Measuring Web Latency and Rendering Performance: Method, Tools, and Longitudinal Dataset. *Transactions on Network and Service Management*, 535-549.

[3] Ali, K.-H. & Lan, S. (2011). Decision Support tools for cloud Migration in the Enterprise. *4th International Conference on Cloud Computing* (pp. 542-548). USA: IEEE.

[4] Bernardita, C. (2022). *12 Cloud Computing Risks & Challenges Businesses Are Facing In These Days.* Berlin: The datapine Blog.

[5] Chnar, M. & Subhi, Z. (2021). Sufficient Comparison Among Cloud Computing Services: IaaS, PaaS, and SaaS: A Review. *International Journal of Science and Business International*, 17-30.

[6] Der-Jiunn Deng, S.-Y. L., Chun-Cheng, L., Shao-Chou, H. & Wei-Bo, C. (2017). Latency Control in Software-Defined. *55*(8).

[7] Dimpi, R. & Rajiv, R. (2014). A Comparative Study of SaaS, PaaS and IaaS in Cloud Computing. *International Journal of Advanced Research in Computer Science and Software Engineering*, 458-461.

[8] Hanan, S., Subhi, Z., Rizgar, Z., Diyar, Z., Omar, A. & Azar, S. (2020). Cloud Computing Virtualization of Resources Allocation for. *Journal of Applied Science and Technology Trends*, 98-105.

[9] Ibrahim, H., Nor, A., Salimah, M., Abaker, I., Abdullah, G. & Samee, K. (2015). The rise of "big data" on cloud computing: Review and open. *Information System*, 98-115.

[10] Jiawei, J., Shaoduo, G., Yue, L., F. W., Gustavo, A., Ana, K. & Wentao, W. (2021). Towards Demystifying Serverless Machine Learning Training. *International Conference on Management of Data* (pp. 857-871). Chaina: SIGMODVirtual Event.

[11] Kalio, Q., & Nwiabu, N. (2019). A Framework for Sceuring Data Warehouse using Hybrid Approach. *International Journal of Computer Science and Mathematical Theory. 5(1)*, 45–56.

[12] Kenneth, G. (2011). Information Management and Evaluation. *Proceedings of the 2nd International Conference on Information* (pp. 90-102). Toronto: Ryerson University.

[13] Lalith, S., Marco, C., Stefan, S. & Anja, F. (2015). Cutting Tail Latency in Cloud Data Stores via Adaptive Replica Selection. *12*(514).

[14] Mamoona, H. (2020). Role of Emerging IoT Big Data and Cloud Computing for Real Time Application. *Computing for Real Time Application*, 495-506.

[15] Michael, L. & carl, G. (2019). Effects of Local Latency on Game Pointing Devices and Game Pointing Tasks. *Conference on Human Factors in Computing Systems* (pp. 1-12). China: ACM journals.

[16] Nassima, T., Olivier, B., Djamal-Eddine, M., Adlen, K., Sohia, A. & Lanon, O. (2020). On Using Physical Programming for Multi-Domain SFC Placement With Limited Visibility. *Transaction on Cloud Computing*, 2787-2803.

[17] Nassima, T., Sophia-Antipolis, Olivier, B., Djamal-Eddine, M. K. & France. (2020). On Using Physical Programming for Multi-Domain SFC Placement With Limited Visibility. *Transaction on cloud computing*, 2787-2803.

[18] Nwiabu, N., Allison, I. & Babs, O. (2021). Modelling Case-Based Reasoning in Situation-Aware Disaster Management. *Communication of the IIMA*, 9(2), 41–66.

[19] Nwiabu, N. & Adeyanju, I. (2012). User Centred Design Approach to Situation Awareness. *International Journal of Computer Applications*, 49(17) 75–88.

[20] Omar, A., Lailan, H., Hanan, S., Rizgar, Z., Sharkir, A. & Moohammad, Z. (2020). Comparison Among Cloud Technologies and Cloud. *Journal of Applied Science and Technology Trend*, 40-47.

[21] Pulkit, M., María, B., Inigo, G., Alvin, L., Willy, Z. & Ricardo, B. (2019). Managing Tail Latency in Datacenter-Scale File Systems Under Production Constraints. *Microsoft Researc* (1-15). Sydney: EPFL.

[22] Rakesh, K., Siba, M. & Chiranjeef, K. (2017). Prioritizing the solution of cloud service selection using integrated MCDM methods under Fuzzy environment. *J Supercomput*, 50-62.

[23] Satyanarayana, S. (2012). CLOUD COMPUTING: SAAS . *GESJ: Computer Science and Telecommunications*, 76-79.

[24] Sherif, S., Anna, L., Daniel, B. M. & Mohammad, A. (2011). A Survey of Large Scale Data Management Approaches in Cloud Environments. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, 311-335.

[25] Suresh, S. & Gopalan, N. (2015). Delay Scheduling Based Replication Scheme for Hadoop Distributed File System. *I.J. Information Technology and Computer Science*, 73-78.

[26] Vanessa, R. (2016). Continuance use intention of cloud computing: Innovativeness and creativity perspectives. *Journal of Business Research*, 1737-1740.

[27] Vinothina, V., Sridaran, R. & PadmavathiGanapathi. (2012). A Survey on Resource Allocation Strategies in Cloud Computing. *International Journal of Advanced Computer Science and Applications*, 97-104.

[28] Xueli, W., Zhigang, Y. & Shiqiang, B. (2012). The Constructive Research on the Evaluation Model of the Industry Cluster Competitiveness. *Engineering Education and Management. 2*. Beijing: International Conference of Engineering Education and Management.

[29] Yongxia, S., Weijin, J., Ying, Y., Haoran, Z., Yirong, J. & Mendeley, A. (2023). Multi-domain authorization and decision-making method of access control in the edge environment. *Computer Networks*, n-pg.

[30] Zaigham, M. (2011). Cloud Computing for Enterprise Architectures: Concepts, Principles and Approaches. *Cloud Computing for Enterprise Architectures*, 3-19.