*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

# Optimizing Cloud Computing Applications with a Data Center Load Balancing Algorithm

**Dr. Nidhi Ranjan[1], Dr. Balasaheb Balkhande[2], Dr. Sanjivani Deokar[3], Torana Kamble[4], Dr. Chaitrali Chaudhari[5], Dr. Shrinivas T. Shirkande[6]**

[1]Associate Professor, Mumbai University, Mumbai, Maharashtra, India
[2]Associate Professor, Vasantdada Patil Pratishthan's College of Engineering & Visual Arts, Mumbai
Mumbai University, Maharashtra, India
[3]Department of Computer Engineering, Lokmanya Tilak College of Engineering, Mumbai University, India
[4]Assistant Professor, Bharati Vidyapeeth College of Engineering, Navi Mumbai, Mumbai University, India
[5]Department of Computer Engineering, Lokmanya Tilak College of Engineering, University of Mumbai, Maharashtra, India
[6]Principal, S.B.Patil College of Engineering Indapur, Pune
nidhipranjan@gmail.com[1], balkhandeakshay@gmail.com[2], sanjivanideokar@gmail.com[3], torana.kamble@gmail.com[4], chaitralichaudhari13@gmail.com[5], shri.shirkande8@gmail.com[6]

**Abstract:**

Delivering scalable and on-demand computing resources to users through the usage of the cloud has become a common paradigm. The issues of effective resource utilisation and application performance optimisation, however, become more pressing as the demand for cloud services rises. In order to ensure efficient resource allocation and improve application performance, load balancing techniques are essential in dispersing incoming network traffic over several servers. The workload balancing in the context of cloud computing, particularly in the Infrastructure as a Service (IaaS) model, continues to be difficult. Due to available virtual machines and the limited resources, efficient job allocation is essential. To prevent prolonged execution delays or machine breakdowns, cloud service providers must maintain excellent performance and avoid overloading or underloading hosts. The importance of task scheduling in load balancing necessitates compliance with Service Level Agreement (SLA) standards established by cloud developers for consumers. The suggested technique takes into account Quality of Service (QoS) job parameters, VM priorities, and resource allocation in order to maximise resource utilisation and improve load balancing. The proposed load balancing method is in line with the results in the body of existing literature by resolving these problems and the current research gap. According to experimental findings, the Dynamic LBA algorithm currently in use is outperformed by an average resource utilisation of 78%. The suggested algorithm also exhibits excellent performance in terms of accelerated Makespan and decreased execution time.

**Keywords:** Load balancing, Resource allocation, Cloud computing, Optimization, task scheduling, SLA

## I. INTRODUCTION

Cloud computing technology has become essential to businesses as the need for internet storage and services increases. It provides many service delivery models, such as web-based software, platforms for creating cloud-based applications, and infrastructure management by Cloud Service Providers (CSPs). The Infrastructure as a Service (IaaS) paradigm, which involves the backend management of data centres, servers, and resource distribution in Cloud Computing technologies, is the subject of this study in particular [1]. As stated in [3], one of the biggest issues in cloud computing is cloud performance. This study's goal is to improve resource distribution, particularly under the Infrastructure as a Service (IaaS) model. Resource distribution is essential for balancing the resources made available to clients and for controlling the workload that user requests place on servers. Users transmit requests, which are modelled by Virtual Machines (VMs), to access services in the cloud [5]. The goal of cloud service providers (CSPs) is to provide users with satisfied services that are beneficial to organisations [6]. As a result, the IaaS model, one of the three service models in cloud computing, which involves managing server workloads, is the primary emphasis of the proposed Load Balancing algorithm. Users can access the frontend portion of a typical cloud environment via an internet connection [7]. In cloud-based systems, incoming user requests are handled via dynamic task scheduling. The required resources are distributed to clients through virtualization, and load balancing is accomplished for the entire system. Virtualization-enabled task scheduling approaches are essential for assuring effective resource allocation and cutting down on execution time [8]. Because virtualization and dynamic task scheduling approaches enable better resource utilisation and higher performance in cloud-based applications, both Cloud Service Providers (CSPs) and cloud consumers can profit from their deployment.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

The way that organisations and people access and use computing resources has been revolutionised by cloud computing. It [2] provides on-demand, scalable services that allow for flexible resource allocation, cost reduction, and cost savings. However, as cloud computing spreads, it is crucial to maximise the functionality and effectiveness of cloud-based applications. The [21] distribution of incoming network traffic among several servers in a data centre is known as load balancing, and it is a crucial component of optimising cloud computing applications. By distributing the load evenly across all servers, load balancing prevents resource overload or underutilization. Maximising application performance, reducing response time, and ensuring equitable resource distribution among users all depend on effective load balancing [9].

As seen in Figure 1 above, task scheduling is vital to workload balancing. Tasks are sent through a cloud broker when users send requests, therefore it's crucial for academics to concentrate on creating an effective algorithm. The suggested approach should successfully distribute tasks across suitable Virtual Machines (VMs) while taking important factors like deadlines into account [10]. This guarantees that a high level of service is maintained and that user requests are carried out and finished within the bounds of the Service Level Agreement (SLA) document's precise specifications [19].

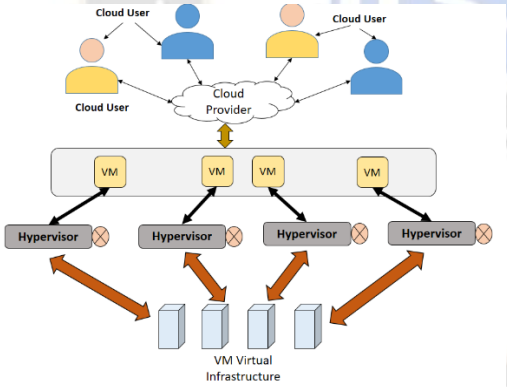


Figure 1: Framework for Infrastructure of Service task Scheduling

User requests must be executed and completed promptly by Cloud Service Providers (CSPs) in order to maintain Quality of Service (QoS) across all delivery models. User requests are delivered over the internet and stored in virtual machines. The effectiveness of the scheduling rules put in place by the Data Broker is crucial to this process. To balance the workload between workstations and servers and achieve best performance and resource utilisation, the scheduling policy must be properly crafted[12].

In this regard, the research is concentrated on creating a data centre load balancing algorithm especially designed for cloud computing application optimisation [18]. By considering variables including server capacity, workload distribution, and real-time performance measurements, the suggested algorithm tries to intelligently distribute incoming requests across servers inside a data centre [12]. The algorithm's goals are to reduce reaction times, increase throughput, and improve application performance as a whole. The system decides how to route incoming requests by dynamically analysing the workload distribution and server capacity, achieving effective load balancing and resource utilisation. Extensive [15] simulations are run in a true-to-life cloud computing environment to assess the performance of the proposed algorithm. The findings reveal how the algorithm stacks up against current load balancing techniques in terms of reaction time, throughput, and resource usage. In order to ensure the algorithm's efficacy in diverse cloud computing environments, its adaptability and robustness in handling dynamic workload fluctuations are also investigated.

**Contribution of Paper:**

- The suggested Load Balancing algorithm seeks to provide high-quality service by efficiently scheduling and balancing workloads while addressing the problem of VM violation in the cloud.
- The Service Level Agreement (SLA) document's key terms, like deadlines, are taken into consideration by the proposed method.
- Proposed method ensure that user requests are executed and completed within the given requirements by effectively allocating jobs to suitable Virtual Machines (VMs) based on these characteristics, hence ensuring a high level of service quality.
- In order to provide load balancing among Virtual Machines (VMs), which is a topic that has not been fully covered by existing methods, the proposed technique additionally involves load migration. The approach efficiently lowers the load balancing parameter and optimises resource use in the cloud environment by combining load migration capabilities.

The rest of the paper is structured as follows. The notion of load balancing and task scheduling is outlined in Section II, along with current research by other authors that highlights its advantages and disadvantages and suggests areas for future research. Details on the proposed LB algorithm are covered in Section III, including the suggested framework, the flowchart, and the pseudocode. The application of the load balancing technique is discussed in Section IV. Section V discusses the load balancing and optimisation results, and section VI concludes with recommendations for the future.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

## II. RELATED WORK

A thorough review of the literature is presented in this section. The notion of load balancing is introduced from the outset, along with its models, metrics, and current standard techniques. This overview lays the groundwork for delving into current work in the area of load balancing, where several techniques put forth by researchers are described and analysed. Also covered are current algorithms that researchers in the field of load balancing have presented.

### A. Task Scheduling and Load Balancing:

As more clients use the cloud, Task Scheduling becomes increasingly important for load balancing. There may be problems with the system's job execution if improper scheduling is used [23]. An effective task scheduling method is necessary to overcome these issues. Manual work distribution is problematic in the cloud context, because users can make use of enormous virtualized resource pools [23]. For big businesses like Google and Amazon, cloud computing services have become indispensable because they ensure uninterrupted data transfer and streaming. However, [20] if the number of customers grows, the underlying algorithms driving these activities can have difficulties and slowdowns. A [17] crucial component of cloud computing is load balancing, which makes sure that user actions are completed quickly and responses are given [24]. Despite its rapid expansion and wide adoption, imbalanced workloads continue to pose problems for cloud service providers (CSPs), which can impair performance and make it difficult to provide consumers with high-quality services.

These problems may involve variables like high Makespan time, which has a negative effect on performance. If the performance of cloud computing applications deteriorates, Service Level Agreements (SLAs), which set forth the expectations between service providers and customers, may be easily broken [25]. As a result of such infractions, the system may become overwhelmed and unable to properly handle incoming tasks, which could lead to task rejection. In order to optimise resource allocation and boost overall system performance in cloud computing environments, task scheduling in load balancing is emphasised as being crucial in the literature. Effective task scheduling becomes increasingly important as the number of customers and the demand for cloud services grow in order to guarantee quick job execution and efficient resource utilisation. The difficulties of poor task scheduling in cloud systems have been emphasised by researchers, particularly when working with massive amounts of virtualized resources, which render manual task allocation impracticable. To [27] overcome these difficulties and increase work scheduling effectiveness, a number of methods and techniques have been presented.

Strong task scheduling algorithms are now necessary to handle rising workloads as a result of the rapid rise of the cloud computing services offered by well-known corporations like Google and Amazon [30]. Poor load balancing and sluggish algorithms can cause task replies to be delayed and performance degradation. Service Level Agreements (SLAs), which outline the expectations between service providers and customers, are threatened by such problems. When the efficiency of cloud computing applications declines, affecting user happiness and system dependability, SLA violations may happen. Researchers [29] have concentrated on creating task scheduling methods that can successfully balance workloads, guarantee high resource utilization, and avoid starving problems in order to address these difficulties. These algorithms attempt to optimise resource allocation, enhance system performance, and provide high-quality services to cloud customers by utilising intelligent scheduling strategies and taking into account elements like job deadlines and completion times [26].

### B. Current Literature:

The authors of [18] provide an algorithm that integrates a three-layer cloud computing network with the load balancing notion. The [28] Opportunistic Load Balancing (OLB) and Load Balance Min-Min (LBMM) approaches are combined in the algorithm. The approach makes use of the ZEUS network framework, which introduces a hierarchical network structure for processing user activities, to improve OLB task scheduling. This approach involves receiving the task, assigning it to the first layer, and then handing it off to a service manager in the second tier. The third level divides requests into more manageable tasks to speed up the processing. In order for the service node to be able to handle the request, assignments are made based on a number of factors, including the amount of available CPU space. This strategy is focused on keeping all nodes active and occupied in order to satisfy user needs [14].

The Improved Load Balanced Min-Min (ELBMM) approach, which aims to maximize resource utilization in cloud computing systems, is provided by the authors in [15]. By choosing the request with the quickest execution time and allocating it to the Virtual Machine (VM) with the quickest completion time, the ELBMM algorithm outperforms the Min-Min algorithm. With this improvement, the utilization cost and system throughput are decreased, enabling more effective resource allocation [35]. The Resource-based Load Balanced Min-Min (RBLMM) approach, which attempts to reduce the Makespan time and accomplish workload balancing among

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

virtual machines (VMs), is proposed by the authors in [18]. The technique calculates the Makespan time as a metric of task completion while taking resource allocation into account. A threshold is established based on this value to direct the work allocation procedure [36].

When compared to the conventional Min-Min algorithm, the results of using the RBLMM algorithm show a considerable reduction in Makespan time of 3 seconds. Although these methods successfully maximise resource usage, they frequently assign jobs in a sequential sequence without taking task or VM priority into account. Additionally, they fail to take into account crucial Quality of Service (QoS) factors for task scheduling, like Deadline and priority. For the advantage of both cloud consumers and service providers, the developers of [27] created effective Scheduling and Load Balancing algorithms that minimise execution time. In order to choose the Virtual Machine (VM) [32] with the lowest cost while accounting for network latency within Data Centres, the proposed method is specifically created. The system automatically determines the ideal data centre with the lowest cost and workload to process the user's request by taking into account workload and cost. This strategy guarantees excellent load balancing and resource allocation, which ultimately shortens execution times and improves the overall performance of cloud services.

In order to improve the overall quality of service, the authors of [21] created a revolutionary Quality of Service (QoS)-based algorithm. The method concentrates on allocation cloudlets with an enhanced balancing technique, resulting in lower Makespan Times for Virtual Machines (VMs) and hosts as well as decreased Completion Time for tasks/cloudlets. The method successfully balances workload and system activity, although it still has flaws. High Makespan values for hosts and virtual machines in particular indicate potential performance problems. Additionally, since the studies were restricted to just three VMs, the algorithm's scalability in large-scale setups remains in doubt. To confirm the algorithm's efficiency and scalability in bigger, more complicated cloud systems, more study and testing is required [31].

Researchers offer a method in [19] that takes user priorities and job length into account. They introduce a credit system that chooses a task in the middle, avoiding the job length distribution's extremes. By figuring out the discrepancy between the duration of the task and the sum of all task requests, credit is given for the middle task. However, this method largely concentrates on work size and disregards crucial quality criteria like Deadline. Although the methodology offers a method for choosing tasks, it fails to take into account important QoS factors, which could have an impact on overall performance and the ability to meet users' demands for timeliness.

A decentralised Load Balancing (LB) technique with an adaptive threshold is presented by the authors in [30]. They put in place a hunger threshold that makes the migration process adhere to a transfer policy. To help with load balancing across VMs, a high latency Virtual Machine (VM) is given a high starvation value. The outcomes show that, when compared to the nature-inspired Honey-bee behaviour baseline method, the suggested STLB algorithm dramatically lowers the number of migrations. This demonstrates how well the STLB algorithm works to achieve load balancing while minimising pointless VM transfers.

## III. PROPOSED METHODOLOGY

The proposed and enhanced load balancing algorithm created for cloud computing settings is presented in this section. Delivering top-notch services to customers in Cloud Computing applications is the main goal of this algorithm. The algorithm includes two crucial procedures:

- Task scheduling: This procedure entails giving individual tasks (cloudlets) due dates and completion times. The method seeks to optimise job execution and ensure timely completion within given deadlines by carefully scheduling tasks based on these characteristics.
- Load balancing procedure: The technique includes workload transfer in cases of Virtual Machine (VM) violations in order to maintain a balanced load in the cloud environment. The technique dynamically redistributes the workload to other available VMs when a VM is overloaded or exceeds specified resource thresholds, ensuring a balanced workload.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
*Article Received:* 27 July 2023 *Revised:* 18 September 2023 *Accepted:* 04 October 2023

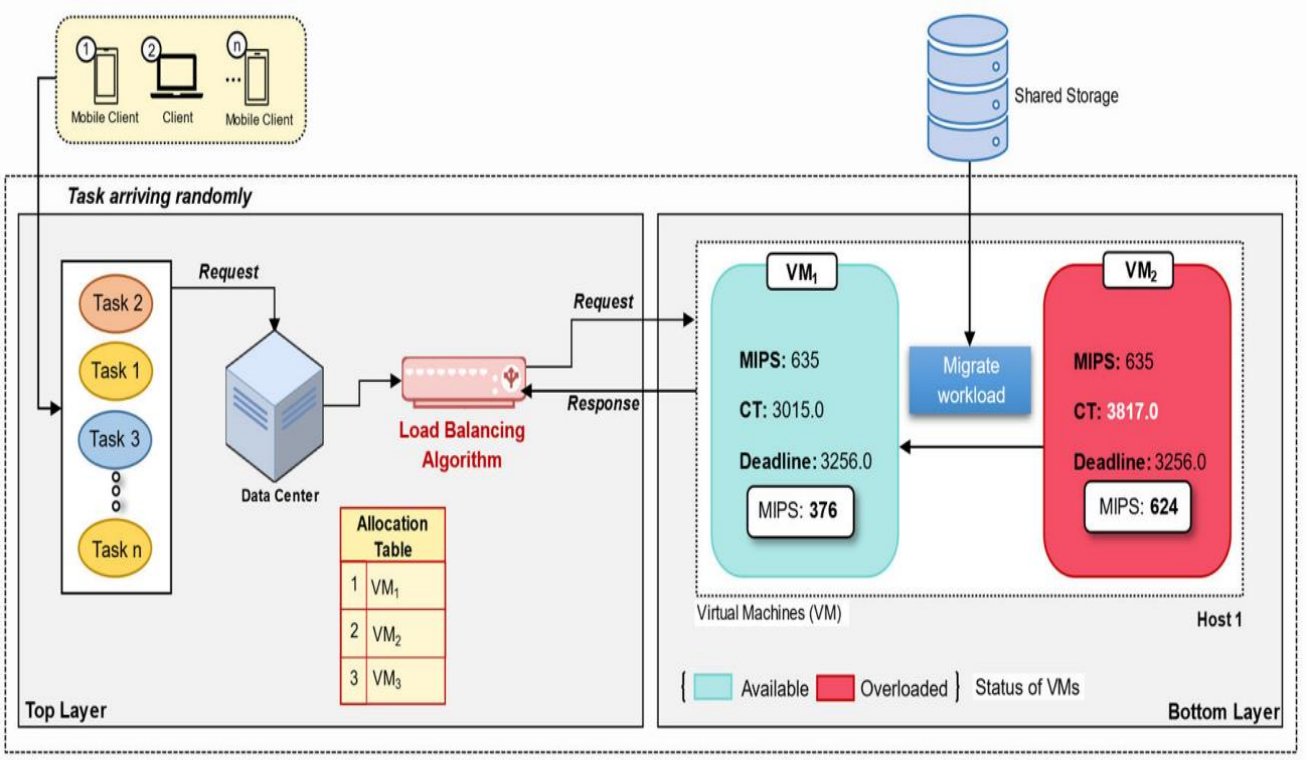


Figure 2: Proposed system for optimization of load balancing

### A. Proposed Work

The suggested approach for this layer addresses requests coming from various clients, including users on mobile devices and desktop computers. These customers use a variety of devices to connect to the Internet and submit their requests to the cloud infrastructure. The Cloudlet Scheduler Time Shared algorithm, which schedules tasks to Virtual Machines (VMs) in a random order based on Arrival Time, is incorporated into the model. The Deadline and Completion Time are the two primary factors taken into account during the job scheduling process. A data centre (DC) in cloud computing is essentially a sizable storage space that houses cloud servers and data. The DC is essential because it takes client requests and sends them to a running load balancer. In order to ensure effective resource utilisation and optimum performance inside the cloud architecture, the load balancer is in charge of evenly distributing the requests across the available resources.

The distribution of requests from users to Virtual Machines (VMs) is the main focus of this layer. We have a primary batch of VMs, as shown in the figure, with VM2 marked as high priority due to a breach of the Service Level Agreement (SLA) requirement. When VM2's completion time exceeds the deadline, this is a violation. The suggested Load Balancing Algorithm (LBA) uses a migration approach to move the burden from VM2 to another available VM in order to address this. Before and after resource allocation, both VMs' MIPS (Million Instructions Per Second) must be reconfigured. Accordingly, the allocation table is updated, displaying the quantity of requests allotted to each Virtual Machine as well as whether or not it is in violation. It's crucial to understand that there is no SLA violation when the Time to Complete (TTC) for a task is shorter than the specified SLA (Deadline). This indicates that the task can be finished by the deadline, guaranteeing that the SLA requirements will be met.

### B. Performance Metrics

Within the cloud context, the suggested Load Balancing (LB) algorithm's performance was assessed using three parameters. The effectiveness of the algorithm was measured and evaluated using the performance measures listed below:

a) Resource Utilization: This measure evaluates how effectively the cloud system uses resources like CPU, memory, and storage. It gauges how well the LB algorithm uses and distributes the resources that are readily available.

$$\mathrm{RU} = \frac{\mathrm{ExT}}{MT}$$

$$\mathrm{RUavg} = \left(\frac{\mathrm{ExT}}{\mathrm{MT}}\right) \times 100$$

b) Makespan: The term "makespan" describes the overall amount of time needed to perform all of the tasks in the cloud system. It gauges how well the LB algorithm manages and arranges the workload by

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
*Article Received: 27 July 2023 Revised: 18 September 2023 Accepted: 04 October 2023*

quantifying the effectiveness of task execution and completion.

$$\text{MT} = \text{Max}(\text{CT})$$

$$\text{MT avg} = \frac{\text{PMax}(\text{CT})}{\text{n}}$$

c) Execution Time: The amount of time required to execute and finish each task within the cloud system is measured by execution time. In an effort to reduce the amount of time needed for job execution, it assesses the effectiveness of the LB algorithm in task scheduling and allocation.

$$ExT = AcT$$

$$\text{ExT avg} = \frac{\text{P(AcT)}}{\text{n}}$$

Insights into the efficiency of the suggested LB algorithm's capacity to maximise resource utilisation, minimise Makespan, and cut down Execution Time in the cloud environment may be gained by examining these three performance indicators.

## IV. PROPOSED LOAD BALANCING ALGORITHM

We describe the suggested Load Balancing (LB) algorithm and the presumptions that were used to develop it in this subsection. The flowchart and pseudocode of the algorithm are also provided to show how it operates. The suggested LB algorithm takes into account both Task Scheduling and Load Balancing features in order to improve the performance of the cloud. By effectively utilising all of the computers' available CPUs and carefully allocating work time, it maximises resource utilisation. The algorithm's main objectives are to minimise Makespan and Execution Time and to maximise resource use.

The proposed algorithm's implementation is predicated on the following premises:

- Multiple CPUs being available: It is expected that the machines included in the cloud infrastructure have multiple CPUs that can be used to execute tasks. As a result, processing can be done in parallel and computer resources can be used effectively.
- Job independence: The method is based on the idea that tasks are independent of one another, so that starting one job does not depend on finishing another. This allows for parallel processing and makes load balancing between the available resources easier.
- Priorities of tasks: The suggested method takes task priorities into account. It gives activities with tougher deadlines or more important requirements a higher priority, ensuring that they are given the proper resources and are completed within the predetermined time limitations.

The phases in the proposed LB algorithm are illustrated by the accompanying pseudocode. Task Length and Deadline, which are crucial components of the Service Level Agreement (SLA) agreement, are represented by random values in the input. The SLA acts as a crucial resource for Cloud Service Providers (CSPs), outlining limitations such work prioritisation and deadline requirements.

**Algorithm 1: Proposed Load Balancing Algorithm**

***Input***: *Task Length, Deadline*

***Output***: *Workload balance among VMs, resource migration in case of SLA violation*

*Step* 1. *Initialize VMs and their resources*

*Step* 2. *Assign initial tasks to VMs based on Task Length and Deadline*

*Step* 3. *while Tasks are available do*:

*a. Calculate the Makespan of each VM*

*b. Calculate the Execution Time for each Task*

*c. Determine the resource utilization for each VM*

*d. Check for SLA violations and prioritize tasks accordingly*

*Step* 4. *if SLA violation is detected then*:

*a. Select the VM with the highest SLA violation*

*b. Migrate resources from the violating VM to another available VM*

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:*** *27 July 2023* ***Revised:*** *18 September 2023* ***Accepted:*** *04 October 2023*

*c. Adjust the resources and workload allocation accordingly*

*Step* 5. *else*:

*a. Continue with the current allocation, as there are no SLA violations*

*Step 6. Output the workload balance among VMs and any resource migration performe*

Workload balance across VMs in the cloud system is the algorithm's primary objective, and it produces resource migration in the event of SLA violations. The LB algorithm attempts to optimise resource utilisation, assure SLA compliance, and accomplish effective load balancing within the cloud environment by following the procedures described and incorporating the relevant computations. In step 5, each Virtual Machine (VM) is given an equal number of MIPs (Million Instructions Per Second). For the host CPU resources to be distributed, MIPs are required. In step 9, each job is given a completion time that is determined by dividing the overall VM length by the MIPS it has been allotted.

Each VM is given a violation fee in step 10 in order to check for SLA violations. By deducting the Deadline from the completion time, this cost is determined. The algorithm determines which VM has the largest violation penalty when there are numerous VMs and gives it a high priority. The method then moves on to modify the CPU resources for that specific VM in the event that there are not enough MIPS on the host to execute it. Additionally, a workload transfer is carried out if the workload from the VM exceeds the host MIPS available. The process flow for load balancing and task scheduling is shown in the diagram. It offers a visual representation of the algorithm's processes, including the decision points and the results the algorithm produced.

The algorithm starts with input values that are below the threshold of 100 GBPS MIPS and below 2000 for the Deadline, such as task workloads from clients. These numbers show how busy the cloud system is. Then, based on the total workload, each Virtual Machine (VM) is given an equal part of the CPU resources. The algorithm assesses whether a VM has broken the SLA requirements by computing the VM cost. This is achieved by keeping track of whether the tasks allocated to the VM take longer to complete than the designated Deadline. The method starts workload transfer to another available VM if a violation is found. By following this procedure, the system workload is maintained in balance and all CPU resources are fully utilised. The technique optimises the utilisation of CPU resources in the cloud system while preserving SLA compliance by dynamically modifying workload allocation and relocating workloads as necessary. This strategy leads to enhanced performance and effective load balancing, which is advantageous to both the clients and the cloud service provider.

## V. RESULT AND DISCUSSION

The goal of the experiment was to show how Makespan, execution time, and resource utilisation could all be improved in a dynamic cloud context. Preemptive task scheduling was considered when the method was being tested. Several Quality of Service (QoS) performance factors for cloudlets are taken into account throughout the scheduling process. These characteristics cover things like the time it takes to do a task, the availability of resources, and SLA adherence. The method makes an effort to optimise job scheduling, ensure effective resource allocation, and achieve the set performance goals within the cloud environment by considering these QoS criteria. The trial confirms the algorithm's ability to give better QoS performance, including decreased Makespan, faster execution, and better resource utilisation.

========== OUTPUT ==========

| Cloudlet ID | STATUS | DC ID | VM ID | Time | Start Time | Finish Time |
|---|---|---|---|---|---|---|
| 8 | SUCCESS | 2 | 3 | 90.01 | 0.1 | 90.11 |
| 24 | SUCCESS | 2 | 6 | 126.05 | 0.1 | 126.15 |
| 16 | SUCCESS | 2 | 5 | 163.14 | 0.1 | 163.24 |
| 14 | SUCCESS | 2 | 4 | 183.28 | 0.1 | 183.38 |
| 15 | SUCCESS | 2 | 4 | 195.09 | 0.1 | 195.19 |
| 21 | SUCCESS | 2 | 6 | 226.25 | 0.1 | 226.35 |
| 3 | SUCCESS | 2 | 1 | 255.81 | 0.1 | 255.91 |
| 11 | SUCCESS | 2 | 3 | 312.27 | 0.1 | 312.37 |
| 10 | SUCCESS | 2 | 3 | 369.06 | 0.1 | 369.16 |

========== OUTPUT ==========

| Cloudlet ID | STATUS | DC ID | VM ID | Time | Start Time | Finish Time |
|---|---|---|---|---|---|---|
| 13 | SUCCESS | 2 | 2 | 97.87 | 0.1 | 97.97 |
| 2 | SUCCESS | 2 | 3 | 124.52 | 2.1 | 126.62 |
| 5 | SUCCESS | 2 | 2 | 173.31 | 10.1 | 183.41 |
| 0 | SUCCESS | 2 | 1 | 246.24 | 0.1 | 246.34 |
| 7 | SUCCESS | 2 | 4 | 254.8 | 11.1 | 265.9 |
| 8 | SUCCESS | 2 | 1 | 416.64 | 3.1 | 419.74 |
| 9 | SUCCESS | 2 | 2 | 457.08 | 1.1 | 458.18 |

Figure 3: Snapshot of Same arrival time and different random arrival time

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

a. Arrival Time: In the CloudSim environment, the cloudlet start time is defined as the cloudlet arrival time, which denotes when the algorithm receives user requests. In CloudSim, by default, every cloudlet arrives at the broker at the same predetermined arrival time. However, adjustments were made in this experiment to bring about alterations in the submission of cloudlets. The experiment attempted to create a dynamic environment where each request may have a varied arrival time by introducing random arrival times. The range of arrival timings makes it possible to create and test algorithms that can efficiently handle and plan jobs in such dynamic environments. Based on the implemented code in this function, the broker then randomly assigns the cloudlets to the Virtual Machines (VMs).

b. Length task: Within the CloudSim system, the size of tasks in bytes plays a critical role in calculating resource usage. Higher resource utilisation rates are often the result of smaller jobs requiring fewer resources. Each Cloudlet in CloudSim has a length value that serves as a type indicator, indicating whether the request is heavy, light, or medium. The length values of the cloudlets were chosen at random for this experiment. By giving each cloudlet a separate value by random assignment, the client requests are set apart from one another. The experiment takes into consideration the variation in task sizes by adding this randomization, allowing for a more thorough analysis of the algorithm's performance under various workload scenarios. The experiment tries to emulate real-world settings where client requests can differ in terms of their compute needs by randomly assigning length values to cloudlets. This method sheds light on the algorithm's reactivity to various task sizes and its capacity to efficiently allocate resources depending on the unique properties of each work.

c. Deadline: The Cloud Service Providers (CSPs) view the deadline, which is the maximum time allotted for a work to be completed, as a critical component of the Service Level Agreement (SLA). Because each client has a different SLA contract based on their particular demands and service expectations from the cloud provider, each Cloudlet in this experiment has a different deadline value. The experiment seeks to imitate a real-world situation where clients have various SLA requirements by giving each Cloudlet a varied deadline value. Instead than employing fixed or consistent deadlines, this dynamic technique instead introduces variability in the deadline values. Due to the fact that it represents SLA commitments, the deadline parameter is extremely important.

Table 1: Summary of result for 10 to 60 task using 4 VM

| No of Cloudlets | Make-span in Average (ms) | Execution Tine in Average (ms) | Utilization of Resources in (%) |
|---|---|---|---|
| 10 | 250.324 | 182.445 | 80 |
| 20 | 512.453 | 345.665 | 74 |
| 30 | 788.578 | 545.221 | 79 |
| 40 | 861.241 | 655.287 | 78 |
| 50 | 973.887 | 745.391 | 72 |
| 60 | 1082.341 | 982.388 | 68 |

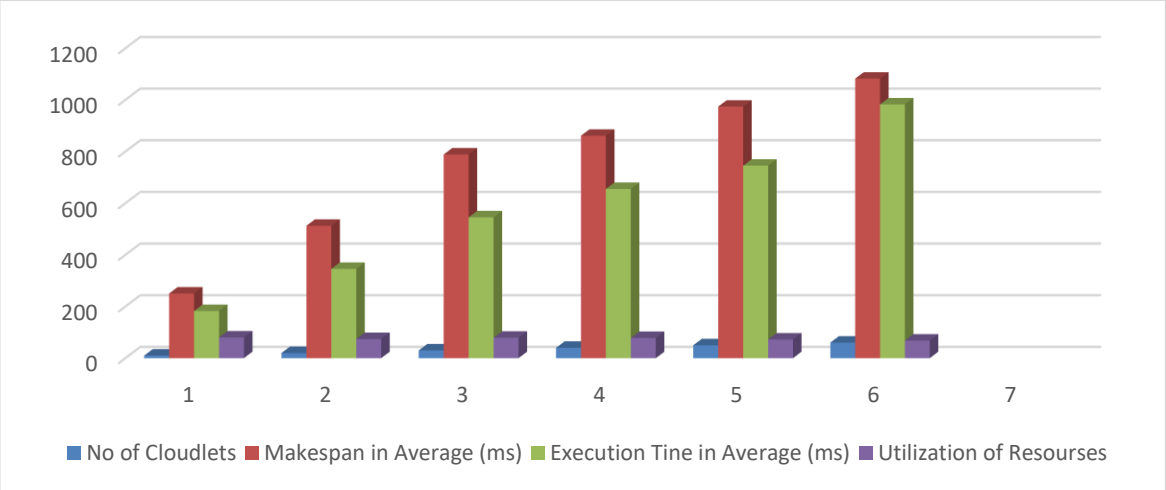


Figure 4: Representation of graph for 10 to 60 task using 4 VM

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

Tables 1, 2, and 3 show the outcomes for various virtual machine (VM) setups when carrying out a variety of jobs ranging from 10 to 60. The makespan, execution time, and resource utilization are the metrics that were examined. The makespan is the amount of time needed to accomplish all tasks. The makespan for the scenario with 4 VMs increases steadily from 10 to 60 cloudlets (tasks) as the number of tasks increases. This shows that as tasks are added, it takes longer to accomplish them all. Additionally, there is a little rise in average execution time, which suggests that for heavier workloads, individual task execution times are slightly longer. The resource utilization, however, stays rather consistent at 80%, indicating that the VMs are being used to their full potential.

Table 2: Summary of result for 10 to 60 task using 8 VM

| No of Cloudlets | Make-span in Average (ms) | Execution Tine in Average (ms) | Utilization of Resources in (%) |
|---|---|---|---|
| 10 | 280.277 | 212.345 | 85 |
| 20 | 562.153 | 383.925 | 78 |
| 30 | 789.778 | 595.739 | 82 |
| 40 | 881.271 | 671.781 | 84 |
| 50 | 993.287 | 775.411 | 80 |
| 60 | 1092.741 | 989.367 | 79 |

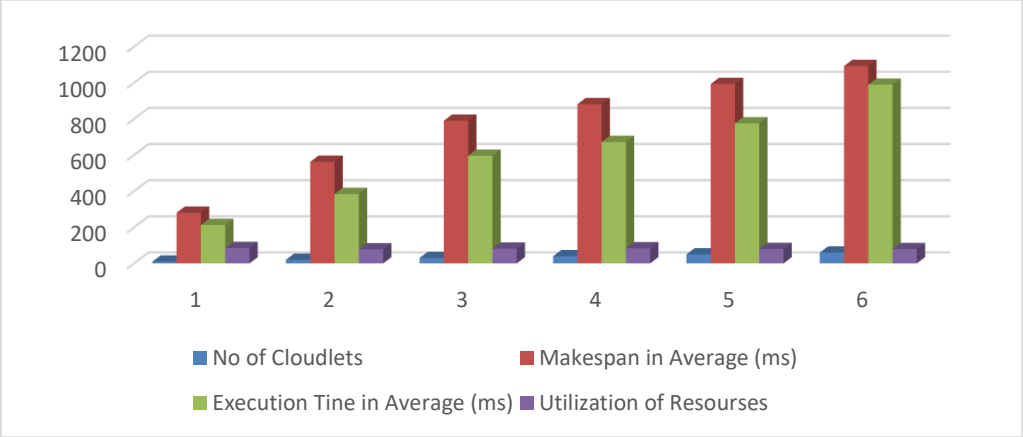


Figure 5: Representation of graph for 10 to 60 task using 8 VM

Similar trends are visible when 8 virtual machines are configured. As the number of cloudlets rises, so do the make-span and average execution time, showing that heavier workloads take longer to finish. Additionally, resource usage rises and reaches about 85%. This implies that the additional VMs aid in more effectively distributing the workload, leading to better resource utilisation.

Table 3: Summary of result for 10 to 60 task using 12 VM

| No of Cloudlets | Make-span in Average (ms) | Execution Tine in Average (ms) | Utilization of Resources in (%) |
|---|---|---|---|
| 10 | 291.217 | 222.145 | 88 |
| 20 | 584.673 | 399.955 | 82 |
| 30 | 791.567 | 605.129 | 85 |
| 40 | 894.622 | 698.181 | 86 |
| 50 | 1003.339 | 778.841 | 82 |
| 60 | 1112.332 | 992.377 | 81 |

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
*Article Received:* 27 July 2023 *Revised:* 18 September 2023 *Accepted:* 04 October 2023

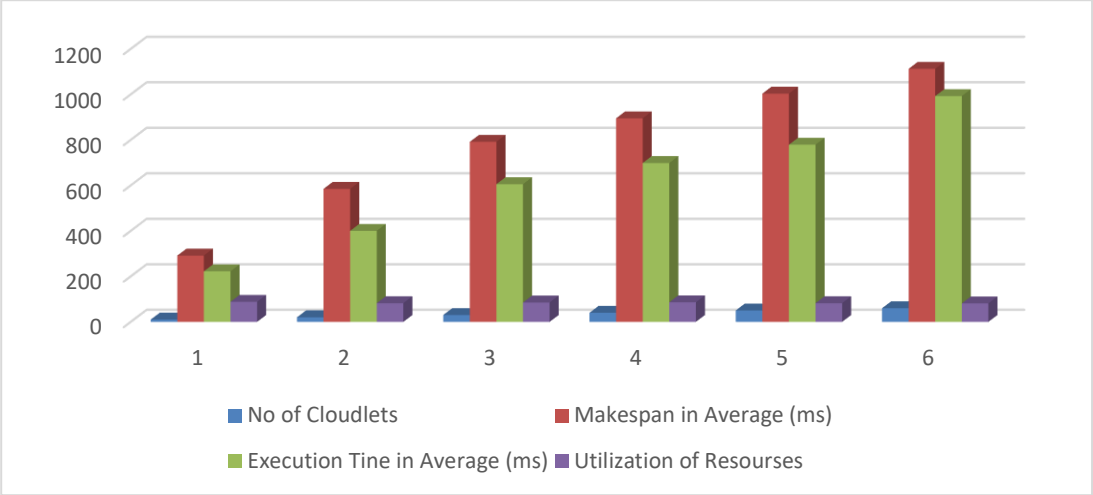


Figure 6: Representation of graph for 10 to 60 task using 12 VM

Finally, the trends are still there when 12 virtual machines are used. As the workload grows, both the makespan and the average execution time exhibit an upward trend. With 60 cloudlets, the average execution time increases and the makespan reaches about 1100 ms. The resource usage increases further and reaches about 88%. This shows that the extra VMs are being used efficiently, which improves performance overall. In terms of makespan and execution time, increasing the number of VMs from 4 to 12 improves performance for heavier workloads. The resource utilisation also rises, showing better use of the resources that are already available. It's crucial to keep in mind that performance gains diminish as workloads increase, indicating that different optimisation methods or scaling tactics may need to be taken into account for even bigger task sets.

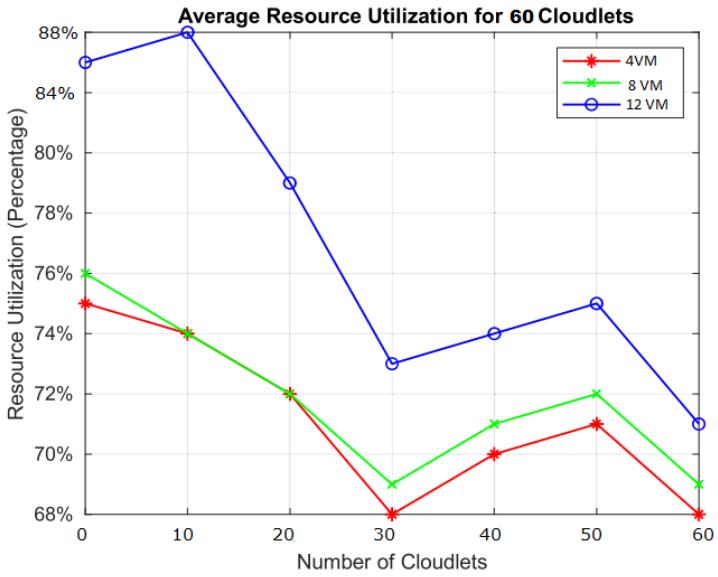


Figure 7: Comparison of 60 task average utilization

Resources in the cloud environment are utilised effectively by the Dynamic LB algorithm. This algorithm was picked as a baseline because it closely resembles the goals of the study and the implementation plan. The researchers intended to take into account QoS parameters or priority as part of their future work on the Dynamic LB algorithm. Our method, in contrast, already includes such parameters to highlight the variations in outcomes when these aspects are taken into consideration. Both methods take into consideration variables like cloudlet length and completion time. But because the Dynamic LB algorithm uses the First Come First Serve (FCFS) technique to schedule jobs, which lacks any kind of prioritisation, it may cause longer waiting times for tasks. However, in order to comply to the SLA agreement and meet QoS requirements, our proposed LB algorithm takes into account various deadlines and arrival times. Through the use of task

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

prioritisation based on each work's unique requirements, this method guarantees improved service in cloud applications.

Our approach seeks to improve overall performance and efficiency of job execution in the cloud environment by using these extra elements in the scheduling process. It offers a more individualized approach that takes into account the unique qualities of each task and prioritises them accordingly, leading to an improvement in service quality and SLA adherence. The Dynamic LB algorithm provides a helpful baseline for comparison because it is applicable to the objectives of the study, but our suggested LB algorithm goes one step further by incorporating QoS parameters and prioritization algorithms. This makes it possible to employ resources in the cloud environment more precisely and efficiently, improving service quality and ensuring that the SLA document's requirements are met.

## VI. CONCLUSION

The proposed technique demonstrates how well it performs in dynamic cloud environments where user requests arrive in a random order and request lengths regularly fluctuate. In addition to handling huge queries, it distinguishes itself from existing methods by doing so. The algorithm takes into account the breach of Virtual Machines' Service Level Agreements (SLAs) by reallocating resources to effectively carry out tasks and reduce SLA violations. The importance of task scheduling in cloud systems for ensuring load balancing and effective resource use. By developing an upgraded load balancing algorithm, this research sought to enhance load balancing. The outcomes showed that, in comparison to the current Dynamic LBA algorithm, our algorithm decreases the make-span and achieves efficient resource utilisation of 88%. In the future, the authors want to improve the performance of cloud-based apps and further optimise cloud resources. To optimise the performance of the method, other SLA parameters will need to be taken into account. To improve overall performance, the algorithm will be checked based on metrics such as the number of SLA violations and the number of migrations. In order to fully assess and validate the suggested algorithm's effectiveness and competitiveness, a complete comparison of it with other algorithms that have already been published in the literature will also be done.

The authors want to continuously improve the algorithm by concentrating on these upcoming projects in order to take advantage of its potential to optimise cloud resource utilisation, improve application performance, and satisfy strict SLA requirements. The continued research and development demonstrates a dedication to improving load balancing methods in cloud systems and expanding the bounds of effective resource management.

## REFERENCES

[1] H. Shukur, S. Zeebaree, R. Zebari, D. Zeebaree, O. Ahmed, and A. Salih, ‘‘Cloud computing virtualization of resources allocation for distributed systems,’’ J. Appl. Sci. Technol. Trends, vol. 1, no. 3, pp. 98–105, Jun. 2020, doi: 10.38094/jastt1331.

[2] M. Agarwal and G. M. Saran Srivastava, ‘‘Cloud computing: A paradigm shift in the way of computing,’’ Int. J. Mod. Educ. Comput. Sci., vol. 9, no. 12, pp. 38–48, Dec. 2017, doi: 10.5815/ijmecs.2017.12.05.

[3] N. Zanoon, ‘‘Toward cloud computing: Security and performance,’’ Int. J. Cloud Comput.: Services Archit., vol. 5, no. vol. 5, nos. 5–6, pp. 17–26, Dec. 2015, doi: 10.5121/ijccsa.2015.5602.

[4] C. T. S. Xue and F. T. W. Xin, ‘‘Benefits and challenges of the adoption of cloud computing in business,’’ Int. J. Cloud Comput.: Services Archit., vol. 6, no. 6, pp. 1–15, Dec. 2016, doi: 10.5121/ijccsa.2016.6601.

[5] D. A. Shafiq, N. Jhanjhi, and A. Abdullah, ‘‘Proposing a load balancing algorithm for the optimization of cloud computing applications,’’ in Proc. 13th Int. Conf. Math., Actuarial Sci., Comput. Sci. Statist. (MACS), Dec. 2019, pp. 1–6, doi: 10.1109/MACS48846.2019.9024785.

[6] S. K. Mishra, B. Sahoo, and P. P. Parida, ‘‘Load balancing in cloud computing: A big picture,’’ J. King Saud Univ.–Comput. Inf. Sci., vol. 32, no. 2, pp. 149–158, 2020, doi: 10.1016/j.jksuci.2018.01.003.

[7] I. Odun-Ayo, M. Ananya, F. Agono, and R. Goddy-Worlu, ‘‘Cloud computing architecture: A critical analysis,’’ in Proc. 18th Int. Conf. Comput. Sci. Appl. (ICCSA), Jul. 2018, pp. 1–7, doi: 10.1109/ICCSA.2018.8439638.

[8] A. Jyoti, M. Shrimali, and R. Mishra, ‘‘Cloud computing and load balancing in cloud computing -survey,’’ in Proc. 9th Int. Conf. Cloud Comput., Data Sci. Eng. (Confluence), Jan. 2019, pp. 51–55, doi: 10.1109/confluence.2019.8776948.

[9] S. H. H. Madni, M. S. Abd Latiff, M. Abdullahi, S. M. Abdulhamid, and M. J. Usman, ‘‘Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment,’’ PLoS ONE, vol. 12, no. 5, May 2017, Art. no. e0176321, doi: 10.1371/journal.pone.0176321.

[10] M. Adhikari and T. Amgoth, ‘‘Heuristic-based load-balancing algorithm for IaaS cloud,’’ Future Gener. Comput. Syst., vol. 81, pp. 156–165, Apr. 2018, doi: 10.1016/j.future.2017.10.035.

[11] B. Singh and G. Singh, ‘‘A study on virtualization and hypervisor in cloud computing,’’ Int. J. Comput. Sci. Mobile Appl., vol. 6, no. 1, pp. 17–22, 2018.

[12] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, ‘‘A comprehensive survey for scheduling techniques in cloud computing,’’ J. Netw. Comput. Appl., vol. 143, pp. 1–33, Oct. 2019, doi: 10.1016/j.jnca.2019.06.006.

[13] F. Zabini, A. Bazzi, B. M. Masini, and R. Verdone, ‘‘Optimal performance versus fairness tradeoff for resource allocation in wireless systems,’’ IEEE Trans. Wireless Commun., vol. 16, no. 4, pp. 2587–2600, Apr. 2017, doi: 10.1109/TWC.2017.2667644.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 10*
*DOI: https://doi.org/10.17762/ijritcc.v11i10.8495*
***Article Received:** 27 July 2023 **Revised:** 18 September 2023 **Accepted:** 04 October 2023*

[14] M. Kumar and S. C. Sharma, ''Dynamic load balancing algorithm to minimize the makespan time and utilize the resources effectively in cloud environment,'' Int. J. Comput. Appl., vol. 42, no. 1, pp. 108–117, Jan. 2020, doi: 10.1080/1206212X.2017.1404823.

[15] G. Patel, R. Mehta, and U. Bhoi, ''Enhanced load balanced min-min algorithm for static meta task scheduling in cloud computing,'' Procedia Comput. Sci., vol. 57, pp. 545–553, 2015, doi: 10.1016/j.procs.2015.07.385.

[16] M. A. Alworafi, A. Dhari, A. A. Al-Hashmi, and A. B. Darem, ''An improved SJF scheduling algorithm in cloud computing environment,'' in Proc. Int. Conf. Electr., Electron., Commun., Comput. Optim. Techn. (ICEECCOT), Dec. 2016, pp. 208–212, doi: 10.1109/ICEECCOT.2016.7955216.

[17] A. V. Lakra and D. K. Yadav, ''Multi-objective tasks scheduling algorithm for cloud computing throughput optimization,'' Procedia Comput. Sci., vol. 48, pp. 107–113, 2015, doi: 10.1016/j.procs.2015.04.158.

[18] B. J. H. Shanthan and L. Arockiam, ''Resource based load balanced min min algorithm (RBLMM) for static meta task scheduling in cloud,'' in Proc. IC-ACT, 2018, pp. 1–5.

[19] A. Thomas, G. Krishnalal, and V. P. Jagathy Raj, ''Credit based scheduling algorithm in cloud computing environment,'' Procedia Comput. Sci., vol. 46, pp. 913–920, 2015, doi: 10.1016/j.procs.2015.02.162.

[20] H. Gamal El Din Hassan Ali, I. A. Saroit, and A. M. Kotb, ''Grouped tasks scheduling algorithm based on QoS in cloud computing network,'' Egyptian Informat. J., vol. 18, no. 1, pp. 11–19, Mar. 2017, doi: 10.1016/j.eij.2016.07.002.

[21] S. Banerjee, M. Adhikari, S. Kar, and U. Biswas, ''Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud,'' Arabian J. Sci. Eng., vol. 40, no. 5, pp. 1409–1425, May 2015, doi: 10.1007/s13369-015-1626-9.

[22] R. Kaur and P. Luthra, ''Load balancing in cloud system using max min and min min algorithm,'' in Proc. Nat. Conf. Emerg. Trends Comput. Technol. NCETCT, vol. 1, 2014, pp. 31–34.

[23] A. Arunarani, D. Manjula, and V. Sugumaran, ''Task scheduling techniques in cloud computing: A literature survey,'' Future Gener. Comput. Syst., vol. 91, pp. 407–415, Feb. 2019, doi: 10.1016/j.future.2018.09.014.

[24] P. Kathalkar, ''Challenges & issues in load balancing in cloud computing,'' Int. J. Res. Appl. Sci. Eng. Technol., vol. 6, no. 4, pp. 963–968, Apr. 2018, doi: 10.22214/ijraset.2018.4163.

[25] S. Afzal and K. Ganesh, ''A taxonomic classification of load balancing metrics: A systematic review,'' in Proc. 33rd Indian Eng. Congr., Jan. 2019, pp. 85–90.

[26] S. Afzal and G. Kavitha, ''Load balancing in cloud computing–A hierarchical taxonomical classification,'' J. Cloud Comput., vol. 8, no. 1, p. 22, 2019, doi: 10.1186/s13677-019-0146-7.

[27] R. K. Naha and M. Othman, ''Cost-aware service brokering and performance sentient load balancing algorithms in the cloud,'' J. Netw. Comput. Appl., vol. 75, pp. 47–57, Nov. 2016, doi: 10.1016/j.jnca.2016.08.018.

[28] Samir N Ajani Piyush K. Ingole , Apeksha V. Sakhare "Modality of Multi-Attribute Decision Making for Network Selection in Heterogeneous Wireless Networks", Ambient Science - National Cave Research and Protection Organization, India,2022, Vol.9, Issue.2, ISSN- 2348 5191. DOI:10.21276/ambi.2022.09.2.ta02

[29] P. Kumar and R. Kumar, ''Issues and challenges of load balancing techniques in cloud computing: A survey,'' ACM Comput. Surv., vol. 51, no. 6, pp. 1–35, Feb. 2019, doi: 10.1145/3281010.

[30] A. Jindal, ''Optimization of task scheduling algorithm through QoS parameters for cloud computing,'' in Proc. ICAET, vol. 57, 2016, pp. 1–4, doi: 10.1051/matecconf/20165702009.

[31] A. Semmoud, M. Hakem, B. Benmammar, and J. Charr, ''Load balancing in cloud computing environments based on adaptive starvation threshold,'' Concurrency Comput., Pract. Exper., vol. 32, no. 11, pp. 1–14, Jun. 2020, doi: 10.1002/cpe.5652.

[32] V. Polepally and K. Shahu Chatrapati, ''Dragonfly optimization and constraint measure-based load balancing in cloud computing,'' Cluster Comput., vol. 22, no. S1, pp. 1099–1111, Jan. 2019, doi: 10.1007/s10586-017- 1056-4.

[33] S.-C. Wang, K.-Q. Yan, W.-P. Liao, and S.-S. Wang, ''Towards a load balancing in a three-level cloud computing network,'' in Proc. 3rd Int. Conf. Comput. Sci. Inf. Technol., Jul. 2010, pp. 108–113, doi: 10.1109/ICCSIT.2010.5563889.

[34] A. Thakur and M. S. Goraya, ''A taxonomic survey on load balancing in cloud,'' J. Netw. Comput. Appl., vol. 98, pp. 43–57, Nov. 2017, doi: 10.1016/j.jnca.2017.08.020.

[35] J. K. Konjaang, F. H. Ayob, and A. Muhammed, ''An optimized max-min scheduling algorithm in cloud computing,'' J. Theor. Appl. Inf. Technol., vol. 95, no. 9, pp. 1916–1926, 2017.

[36] A. Dhari and K. I. Arif, ''An efficient load balancing scheme for cloud computing,'' Indian J. Sci. Technol., vol. 10, no. 11, pp. 1–8, Mar. 2017, doi: 10.17485/ijst/2017/v10i11/110107.