Wireless Personal Communications (2024) 137:2145–2165
https://doi.org/10.1007/s11277-024-11479-4

A Critical Review and Analysis of Load Balancing Methods in Cloud Computing Environment

Anjali Choudhary[1] · Ranjit Rajak[1] · Shiv Prakash[2]

Accepted: 15 July 2024 / Published online: 6 August 2024

Abstract
In the present times, the concept of cloud computing has played a significant role at the global level. With this approach, users can able to customize their services as per their needs. By having the connection of internet users can get able to serve various kinds of services like on-demand access, storage space, software building platforms, data recovery, etc., and pay only for that service that they have consumed. Enormous challenges in the cloud domain such as fault tolerance, energy efficiency, scheduling, resource provisioning, load balancing, etc. This paper is focused on load balancing domain. This can be defined as a redistribution of the workload among various available virtual machines in such an identical manner that would lead to a balanced state. This paper presents the evaluative and inclusive review of numerous load balancing (LB) methods. Quality of services(QoS) is vital role that contain various parameters to evaluate the load balancing methods in respect of makespan, speedup, cost, throughput, etc. This paper is highlighted numerous of load balancing methods with their brief explaination, platform used, different simulator and tools used by these methods and based on QoS parameters.

Keywords Cloud computing · Load balancing · QoS parameters · Cloudsim · Makespane

1 Introduction

There are various emerging trends in computer science fields and it is changing as per market demands, complex applications, and user-friendly requirements. Cloud computing is one of the latest trends in IT industries and the fastest growing technology due to its applications in many domains such as Military Science, Advanced Quantum machines, Management Science, E-Commerce, different biological sciences for data simulation, and many

✉ Ranjit Rajak
ranjit.jnu@gmail.com

[1] Department of Computer Science and Applications, Dr. HarisinghGour Central University, Sagar, India

[2] Allahabad University, Prayagraj, India

more other different subject areas. Formally, it is defined as the platform for end users where the Internet should be the active connection and provide the user requirements in terms of resources both hardware and software. It is an extension of conventional systems such as distributed, grid, and parallel computing [1] and it works on the principle [2] "*Pay and Use the Resource"*.According to NIST [3], the definition of cloud computing is "Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with the minimum management effort or service provider interaction". Cloud computing is equivalent to distributed computing over a network and can execute multiple programs or applications concurrently. Two major focuses [4] of the cloud computing platform are *Virtualization and Abstraction.* Two major classifieds of cloud computing models are deployment models and service models [5] as shown in Fig. 1. Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) belong to the service model category. Similarly, public, private, community, and hybrid clouds are belongs to the deployment models categories. This computing is prominent technology but there are several issues/challenges [6] that have to be concerned. Some of them are *performance*, *bandwidth cost*, *portability and interoperability*, *availability and reliability*, *scalability and elasticity*, *what to migrate*, *virtualization*, *security and privacy*, *service delivery and billing*, etc.

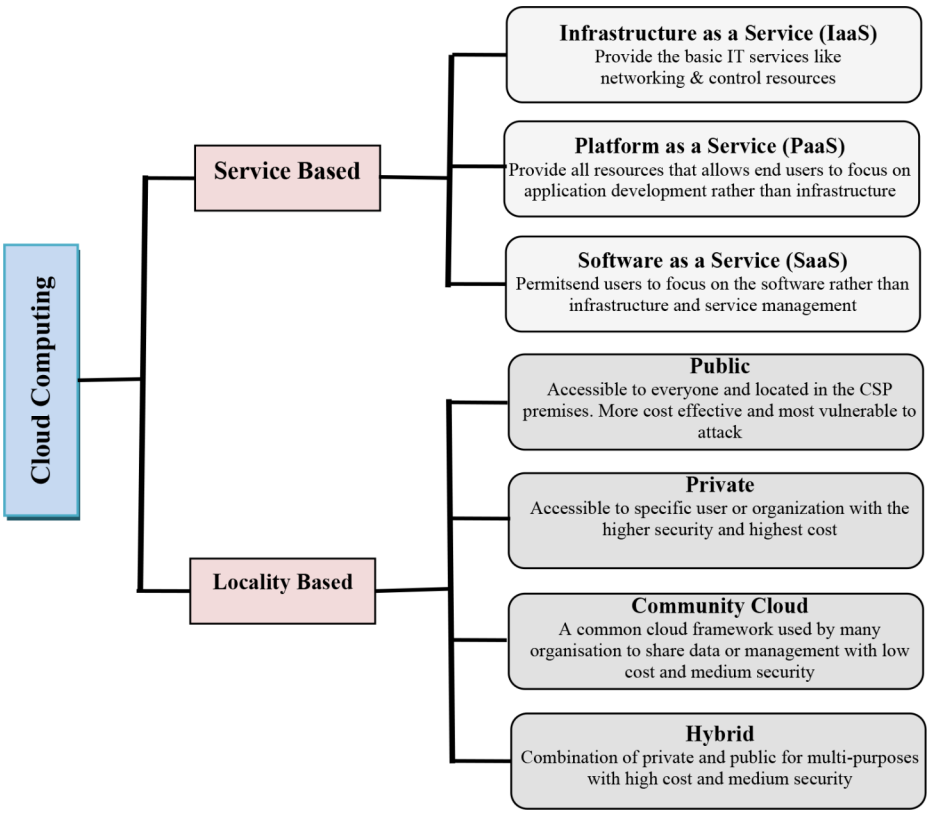

Fig. 1 Cloud computing categorization [7]

Cloud Computing can be categorized into three characteristics [9]: first is *cloud computing works based on the virtualization technology to access services and to realize fast deployment of resources*, *second is user can get cloud computing facilities framed services that are designed to the huge amount of information and get accessed via the internet*, *and the third is cloud computing resources can be dynamically extended and customize according to the needs of users and charged based on what they have used only*. An important fact is that users don't need to control and manage them individually, which can decrease the burden of the end user's dependence on IT expertise and processing.

General framework of cloud computing shown in Fig. 2 which is consisting five layers such as *User Infrastructure Layer*, *Cloud Application Layer*, *Platform Layer*, *Unified Resource Layer*, and *Infrastructure Layer*. These layers are fundamental for designing the cloud domain [8] and the details of these layers are as follows:

- *User Infrastructure Layer*: This layer represents the front end of cloud users. And it contains the tablet, PC, mobile devices and servers, etc., and is used by end users/cloud users to access the services of the cloud computing system. Here Internet plays a vital role because cloud users get connected via the internet to the cloud.
- *Cloud Application Layer*: This layer is related to application software or resources are available to the cloud users in a direct way. CSP (Cloud Server Provider) collect these resources and deploy them based on an 'on-demand access or 'pay as you go, model. Here applications like customer relationship management (CRM) e–services and e-research [8] can be deployed by cloud users.
- *Platform Layer*: This layer provides the platform level services and delivers development, hosting, deployment and managing. All these services at the user level. Major services are resource management, load balancing, scheduling and service discovery.
- *Unified Resource Layer*: This layer contains pods, virtual machines, and logical storage, Physical resource layer is abstracted by this layer.
- *Infrastructure Layer*: This layer comprises physical resources that contain huge number of servers/ host machines and physical storage devices. All these depend on the size of the cloud data center.

Several challenges require to be solved such as [6] privacy, legal, vendor lock-in, open standards, security, IT Governance, consumer and storage, performance interference and noisy neighbors, load balancing, etc.

2 Load Balancing in Cloud Computing

Cloud computing is trending and its users are growing at rapid speed which is also increasing the load on the resources such as virtual machine and some others etc. The term load balancing is associated with load on the resources which provides a way to balance the load among the machine which are overloaded, idle, or underloaded.

It is the consistent distribution of load among available machines where the load can be defined as the complete or total allotment of the task to the virtual machine. A noteworthy load balancing approach not only boosts the overall performance but also enhances the performance, matrices also called quality of services (QoS) i.e. makespan, response

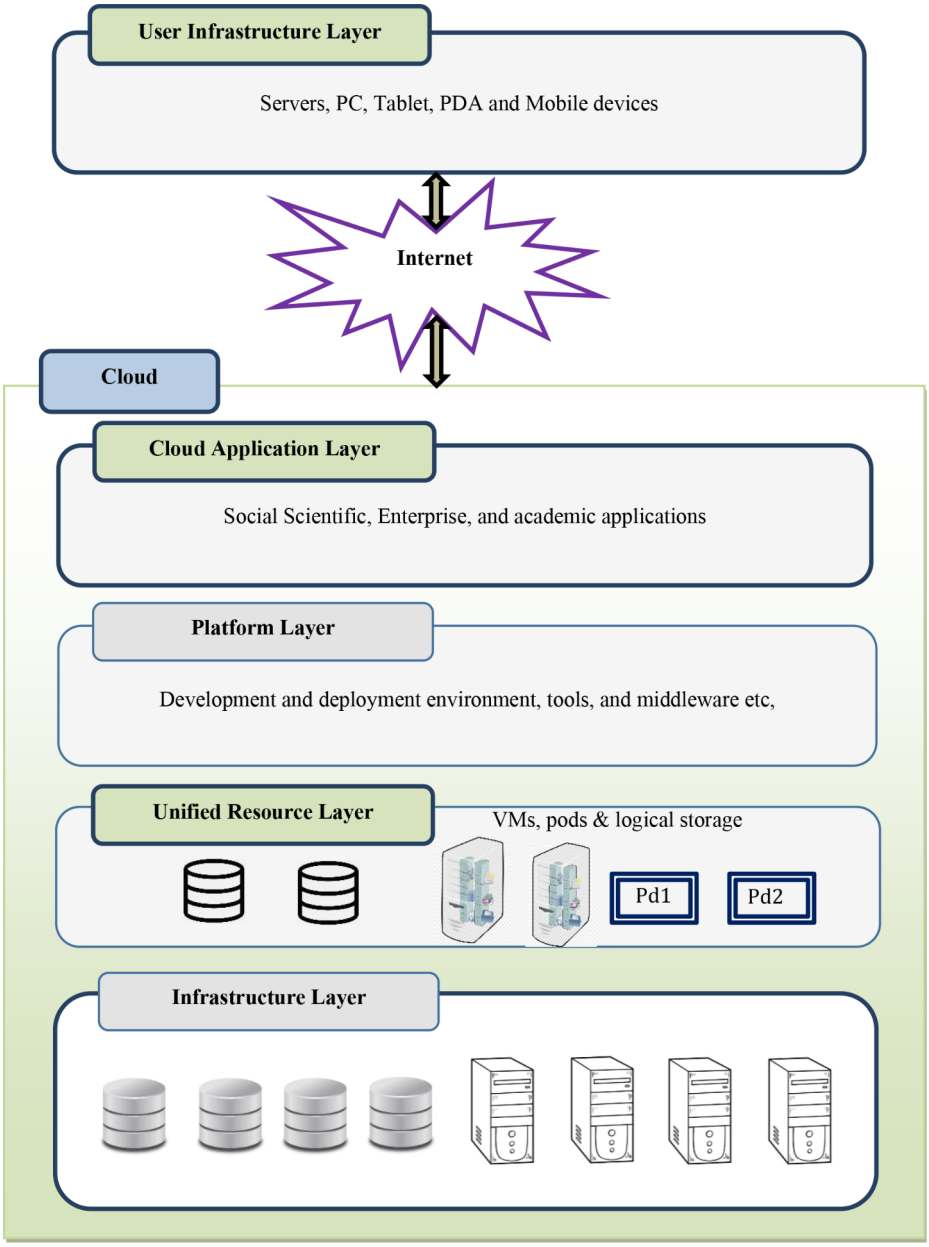

Fig. 2 Cloud architecture [8]

time, resource utilization, throughput, etc. Load balancing leads to better user satisfaction and ensures that cloud resources are highly utilized. It is also provides a massive improvement in performance and maintains system stability in adverse situations like system failure, overwhelming loads of machines, etc.

Hardware load balancing and software level balancing are two classified of LB. First is hardware load balancing and another is software load balancing. Hardware-based load balancing can be done by a hardware load balancer that is placed in the forepart of the server and provides the direction of all requests to the server. This direction of the path is based on the performance of the system i.e. utilization of memory, CPU, and VMs. Here is a routing manager that directs and distributes server load based on server resources. Whereas software based load balancing, the service executes on each machine in a clump or cluster, if any one of them goes down or fails, another machine in the clump can take place and alter the communicating path among the available machines and involve to engage the extra load, here server the server machine helps to remove the single point of failure of a clump/cluster.

The basic architecture and working process of load balancing in the cloud domain as shown in Fig. 3 [11].

As shown in the above architecture of LB, all the end users submit the request to the virtual machine (VM) for accessing their task on the application window with the help of the Internet. After that cloud service provider (CSP) collects all request and pass them to the cloud manager. Cloud manager associated with load balancer, here load balancer computes all the details regarding the status of unassigned VM by the efficient use of load balancing algorithms. Based on this information load balancer identifies the idle, overloaded, or underloaded status of the VM and then distributes the load among all the available VM accordingly.

2.1 Classification of Load Balancing(LB) Approaches

This section discusses the classification [27] of LB which is based on two factors such as System Load and System Topology. The details of classification shown in Fig. 4.

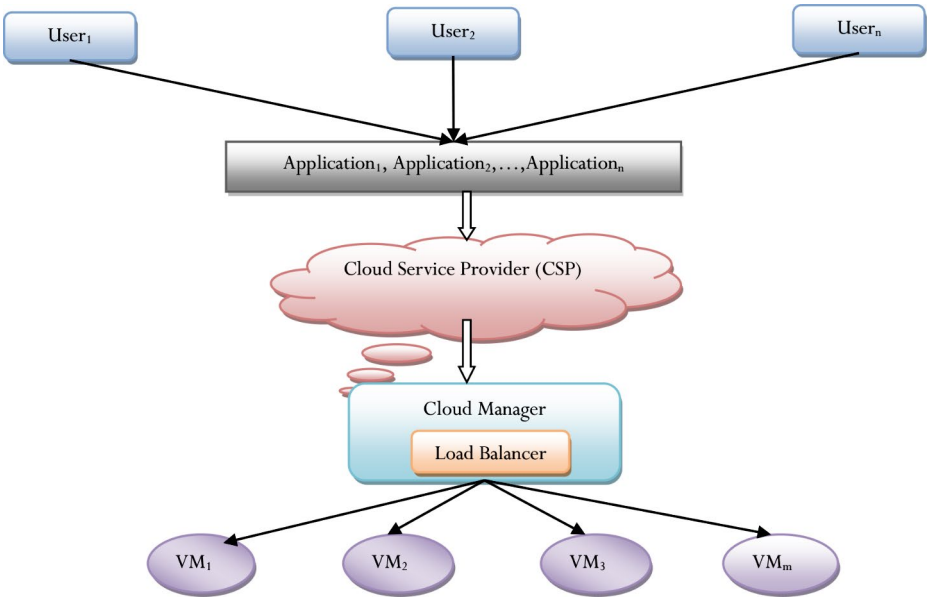

Fig. 3 Basic architecture of load balancing

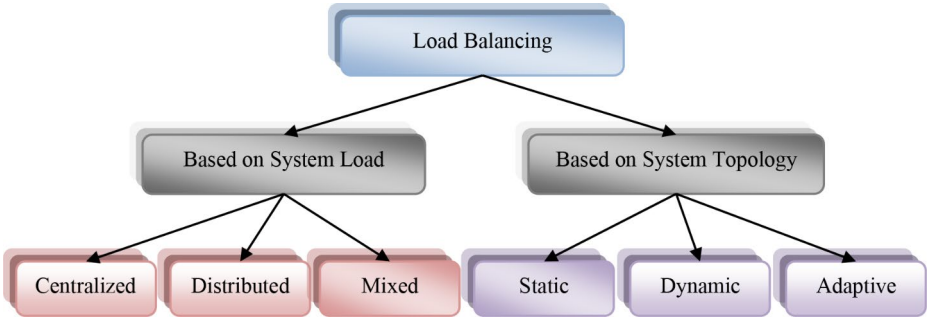

Fig. 4 Classifications of load balancing approaches

- *System Load*: This load balancing is classified into centralized, distributed and mixed approaches. Details of these approaches as follows:

  Centralized Approach: It is the central part of the whole network which is control and manages the allocation of resources.
  Distributed Approach: Here, each node collects load information from other nodes and then creates its load vector independently. These local load vectors are responsible for the local decisions.
  Mixed Approach: It is a combination of both centralized and distributed approaches and singly provides benefits.

- *System Topology*:This load balancing is also classified into three parts such as static approach, dynamic approach and adaptive approach. Details of these approaches as follows:

  Static Approach: Homogeneous and stable environment uses a static algorithm and delivers a better result. If during the execution time dynamic changes to the attributes occur then this approach does not meet the requirements and flexibility.
  Dynamic Approach: During before and during execution time, both conditions are desirable for the dynamic approach because it is more flexible and also able to take a different kinds of attributes in the system.
  Adaptive Approach: In cloud computing, when the system changes are done frequently, the adaptive approach provides better performance.

This kind of approach does the satisfactory distribution of system load by converting. their parameters dynamically and also their algorithms.

2.2 Challenges of Load Balancing

There are two important parties in the cloud computing environment which are cloud service providers (CSP) and end-user who are using cloud resources. Various challenges [28] are shown in Fig. 5 faced by these parties during using cloud services. The brief details of these challenges are in Table 1.

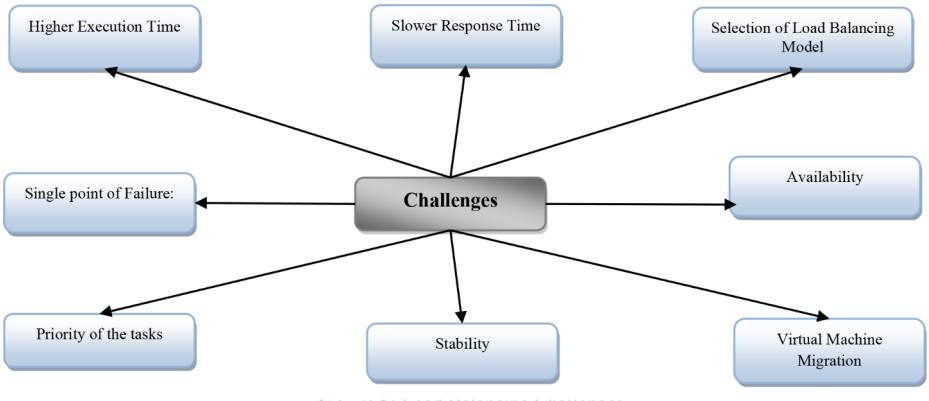

Fig. 5 Load balancing challenges

3 Quality of Service(QoS) Parameters

This section discusses various parameters [12] that affect load balancing in the cloud computing environment. These parameters are used to allocate the tasks onto the virtual machine and also help to analyze the performance between the various models in the cloud computing environment. There are two major parameters such as makespan and energy consumption. Other parameters are reliability, fault tolerance, associated cost, migration time, response time, throughput, thrashing, accuracy, scalability, and predictability. These parameters are also known as Quality of Services (QoS) parameters which can be expressed as follows:

I. ***Turnaround Time (TT)*** [13]: It is defined as the execution of the tasks of the given workflow on the cloud platform and the duration of time between submission time and finishing time. It is formulated as.

$$TT = Max\{AFT_{i,j}\} \tag{1}$$

II. ***Actual Finishing Time(AFT)*** [13]: AFT of task Ci on virtual machine Vj is computed as the earliest start time (EST) of C_i and $EFT_{i,j}$ and formulated as.

$$AFT_{i,j} = EST + \frac{C_i^{length}}{V_j^{cap}} \tag{2}$$

Where EST is the estimated start time of task C_i.
C_i^{length} is the length of task C_i.
V_j^{cap} is the capacity of the virtual machine V_j.

III. ***Average Response Time (RT_{avg})*** [13]: The difference between the earliest start time (EST_i) and arrival time(AT_i) provide response time and RT_{avg} of all task can be calculated as.

Table 1 Details of challenges in load balancing

S. No.	Challenges	Brief Explanations
1	Higher Execution Time	• Due to the geographical distribution [5] of cloud nodes, they performed efficiently in cloud areas if it is for closely located nodes. • As their location of them is increased, there are some factors like communication delays, and network bandwidth in network topologies are effected the execution time and take it to a higher position.
2	Slower Response Time	• This issue is associated with the previous one. • The geographical distribution of large distances between nodes leads to higher execution time, and it is responsible for the slow response.
3	Selection of Load balancing Model	• In cloud computing technology selecting a suitable load-balancing model is a very tough task. • whereas several models are available like BRS, VD, CAB, CARTON, ACCLB, MmLB, etc. [7].
4	Virtual Machine Migration	• The physical machine becomes overloaded due to migrating different VM structures [11]. • Because one physical machine supports multiple virtual machines.
5	Priority of the task	• Due to execution time, users have different requirements for the task. • If the user wants to perform the priority-based task so there should be a heterogeneous scheduler that can define the priority of the task.
6	Single Point of Failure	• All the tasks are assigned by the central node to the other nodes. • if the central node gets fails then the entire system will fail [11].
7	Availability	• Availability ensures that clients get services in 24×7 time.
8	Stability	• System stability is a necessary part of the load balancer. • It should be stable when the system load increased suddenly.

$$RT_i = EST_i - AT_i \tag{3}$$

$$RT_{avg} = \frac{(\sum_{i=1}^{n} RT_i)}{n} \tag{4}$$

IV. ***Average Utilization (U_{avg})*** [13]: Utilization of each VM can be defined as the utilization of the system. Resource utilization is one of the major factor for associated Cost (C_a). It can be computed as:

$$U_j = \frac{MAT_j}{TT} \tag{5}$$

Where MAT_j refers last computed machine available time of V_j and TT is turnaround time.

The average utilization can be calculated as:

$$U_{avg} = \frac{(\sum_{j=1}^{m} U_j)}{m} \tag{6}$$

V. ***Migration Cost (MIG$_{cost}$)*** [14]: It is time elapses for the migration of various resources from one host to another host. This migration can be either task or a VM among the host machines. Maximum M_t will degrade the load balancing and makespan. It is one of the major factors is the VM crashing during the execution of the task. The migration cost of task Ci is computed as the task transfer from one virtual machine to another and denoted as.

$$MIG_{cost} = \frac{C_i^{size}}{BW_k} \tag{7}$$

Where C_i^{size} is the size of task C_i and $BW_k$is the bandwidth of k^{th} virtual machines.

VI. ***Estimated Computation Time(EC)*** [15, 16–18]: It is the scheduling attributes used for allotments of task C_i onto a virtual machine and computed as follows.

$$ECT_{ij} = \begin{bmatrix} ECT_{11} & ECT_{12} \cdots & ECT_{1n} \\ ECT_{21} & ECT_{22} \quad \ldots . & ECT_{2n} \\ ECT_{m1} & ECT_{m2} & ECT_{mn} \end{bmatrix} \tag{8}$$

Where ECT_{ij}isthe time of job/task C_i on virtual machineV_j.

VII. ***Average ECT (ECT$_{avg}$)*** [13, 15]: Average ECT of task $C_i$can be calculated as the ratio of the summation of ECT of all machines and the total number of machines. i.e.

$$ECT_{avg} = \frac{\sum_{j=1}^{TotalM} ECT_{i,j}}{M_{total}} \tag{9}$$

VIII. ***Critical Path(CP)*** [15, 19, 20]: Critical path of the C_i^{entry}*to* $C_j^{exit}$is computed as.

$$CP = \max_{path \in C_i} \{length\,(path)\} \tag{10}$$

$$where\ length\,(path) = \sum_{C_i \in C} ECT_{avg}\,(C_i) \ + \sum_{e \in E} D_T\,(C_i, C_j) \tag{11}$$

IX. ***Earliest Start Time (EST)*** [15, 21]: It is defined as follows:

$$EST\left(C_i, V_j\right) = \left\{ \begin{array}{ll} 0 & if C_i \in C_{entry} \\ \max\limits_{C_i \in pred(C_i)} \{EFT(C_j, V_j) + MET\left(C_i\right) + D_T\left(C_i, C_j\right)\} & otherwise \end{array} \right\} \quad (12)$$

X. ***Minimum Execution Time (MET)*** [15, 21] It is calculated as follows:

$$MET\left(C_i\right) = min.\left\{ECT\left(C_i, V_m\right)\right\} \quad (13)$$

XI. ***Earliest Finished Time (EFT)*** [15, 22]: It is computed as follows:

$$EFT\left(C_i, V_j\right) = ECT_{ij} + EST\left(C_i, V_j\right) \quad (14)$$

XII. ***Load Balance Level*** [13]: The load balancing level can be defined as the variation of utilization time on an available virtual machine from average utilization and calculated as.

$$LB_l = \sqrt{\sum\nolimits_{j=1}^{m} \frac{\left(U_j - U_{avg}\right)^2}{m}} \quad (15)$$

XIII. ***Other Cost-related Parameters*** [14]: Total gain represents an economic aspect of the core concern for the cloud users of the system and cloud service provider and is related to cost calculated as:

$$TG = \sum\nolimits_{i=1}^{n} Profit_i - \sum\nolimits_{i=1}^{n} Loss_i \quad (16)$$

$$TL = \sum\nolimits_{i=1}^{n} Loss_i - \sum\nolimits_{i=1}^{n} Profit_i s_i \quad (17)$$

Where TG is Total Gain and TL is Total Loss

$$Profit_i = (Deadline\ of\ C_i - AFT_{i,j}\ ) * Cost\ of\ V_j \quad (18)$$

$$Loss_i = (AFT_{i,j}\ - Deadline\ of\ C_i) * Cost\ of\ V_j \quad (19)$$

XIV. **Makespan(M_s)** [17, 20]: It is defined as the total time taken by a job for its completion from start to end state. It should be a minimum. All tasks execution finish on available VM and computed as:

$$M_s\ = Min.\left\{EFT(\ C_{exit},\ M\ )\right\} \quad (20)$$

XV. **Waiting Time(W_t)** [12]: Waiting time is refers to the total amount of time the ready task waits for the CPU to be assigned.

$$Wt = TAT - BT \tag{21}$$

Where TAT stands for turnaround time and is calculated as the difference between completion time and arrival time.
BT refers to burst time and is calculated as the net amount of time taken by the CPU to execute the whole task,

XVI. **Response Time(R_p)** [24]: Efficient makespan depends on response time. Simply acknowledged for a task by the system when the task is submitted to the system. Integration of transmission time (T_t), service time (S_t), and waiting time(W_t) of task(t) in the system. i.e.

$$R_p(t) = T_t + S_t + W_t \tag{22}$$

XVII. **Reliability (R_e)** [12]: Major factor of reliability is system configuration. It can be improved by the availability of other resources in case of failure of any system during the execution of the job. The stability of any system also depends on the reliability metric.

XVIII. **Throughput(T_p)** [26]: Number of tasks executed per unit time by resource i.e. VM. System performance is measured in terms of throughput and it should be maximized. It is calculated as:

$$T_p \alpha \frac{1}{M} \tag{23}$$

XIX. **Thrashing(T_s)** [12]: It is related to system resources such as memory i.e. paging system. In respect of cloud computing, VMs are taking more time in migration rather than executing the task. It disturbed the proper scheduling of the tasks on VM.

XX. **Accuracy(A_{cc})** [12]: Corrective result of execution of the task.The important factor for today's technology world. It minor decreases the makespan.

XXI. **Predictability(P_b)** [12]: It is an important factor for load balancing and makespan. It decides how to the allocation of tasks, execution of the task, and completion of the task on available virtual machines.

XXII. **Scalability(S_c)** [12]: It is an important feature of any computer system. Define the capacity of the system in case of overload due to the size of the task or the increased number of tasks in the system.

XXIII. **Energy Consumption (E_c)** [29]: It is also an important metric for cloud computing. The primary objective is to minimize energy consumption. Major resources for energy consumption are personal terminals, networking nodes, and local servers.

XXIV. **Fault Tolerance(F_t)** [12]: F_t is one factor for system capability and mechanism to provide regular services in case of one or more system elements are failed. It is a little bit costly.

XXV. **Speedup(S_{speed})** [23, 24, 25]: It is calculated as the total value of minimum ECT of all the task Ci and the scheduling length of a given DAG on a Vj.

$$S_{speed} = \frac{Min[\sum_{j=1}^{m} ECT_{i,j}]}{scheduling\ length} \tag{24}$$

4 Critical Study and Analysis of Load Balancing Methods

This section is the study of load balancing methods it is divided into four subsections such as the brief explanation of LB methods, based on the simulator and tools used, based on environments, and based on evaluative parameters. There are twenty load-balancing methods are taken for this study.

4.1 Load Balancing Methods

This section included twenty load balancing methods including the name of the article, it designated all algorithms by load balancing method (LBM) i.e. LBM_1,LBM_2….LBM_{20}, and it also includes a brief explanation of all twenty algorithms as given in Table 2.

4.2 Simulator Tools & Techniques

Simulator tools are the core solution in the process of deployment. The Performance of the application can be determined by the simulator. By using these tools users and service providers can get performance reports of the relevant application and can get this service at no cost. Some most popular simulators are discussed here:

a. **Cloudsim**: University of Melbourne, Australia bringsforthacloudsimsimulator [10]. This simulator can be accessed as an open source and can work with Unix/Linux and windows. In 2009, the first version of cloudsim has been launched, and another version came into the market with an advanced version.
b. **Cloud Analyst**: It is an extended version of the cloudsim simulator. It provides a high level of flexibility and configuration in simulation [49]. Simulation can also be saved in different file formats [50]. Changes in parameters can be done easily without focusing on coding and executing repetitively for the same parameter or altered manner.
c. **Grid Sim**: It is another simulator tool for simulation. It is based on SimJava2 [51] and it is a general-purpose discrete event simulation package that is carried out in java. It uses a message-passing operation elucidated by SimJava to communicate with each other among components.
d. **Matlab**: It is a high-level programming language used by scientists, researchers, and engineers to perform the calculation of array mathematics and matrix directly. Matlab can be used to run from a simple program to a complex one. The desired simulator can be created with the use of Matlab because of its large availability of functions and tools. This kind of simulator provides better user requirements to execute their codes and find appropriate results. Nazi TabatabaeiYazdi et al. used Matlab to create a multicloud simulator to implement their experiment [52].
e. **Mininet**: Mininet is an emulator that is used to test how software is interconnected while underlying hardware or hardware and software are together. Minimet [53, 54] is

Table 2 Summary of LB algorithm

Name of article	Year	Designated by	Brief Description
Join-Idle-Queue: a novel load balancing algorithm for dynamically scalable web services [29],	2011	LBM1	• Proposed JIQ (Join Idle Queue), a novel class algorithm that use to distribute balance load in a large system. • It is based on two levels of load balancing approach. • A data structure called queues communicates with this approach. • The initial level of LB in JIQ works and delivers the best possible server for task allotment with consideration of the critical path. Further than the allotment of the best possible service to the I queue done on the second level of LB.
Task-based system load balancing in cloud computing using particle swarm optimization [30]	2013	LBM 2	• Proposed a TBSLB-PSO algorithm that attains system load balancing by transmitting only additional tasks from an overloaded state of the VM as an alternative to whole overloaded VM migration. • Design a model of optimization to migrate additional tasks to the new host VM with PSO.
Genetic algorithm and gravitational emulation-based hybrid load balancing strategy in cloud computing [31]	2015	LBM 3	• Introduced a novel LB algorithm to search underloaded nodes and balanced them as per their load • Proposed algorithm compared with traditional algorithm.
Load balancing in a cloud computing environment based on an improved particle swarm optimization [33]	2015	LBM4	• Proposed an algorithm to attain the optimization of resource LB in the cloud environment. • It takes the attributes of complex networks and set up a correlating resource task allocation model.
Proposing a load balancing method based on cuckoo optimization algorithm for energy management in cloud computing infrastructures [35]	2015	LBM 5	• Introduced a technique based on the cuckoo search optimization algorithm. • Detect overutilized hosts. • Implement the MMT (minimum migration policy) for the migration of VMs.
A multi stage load balancing technique for cloud environment [36]	2016	LBM 6	• Introduced two levels of load balancing approach by collaborating join the idle queue (JIQ) and join shortest queue (JSQ) approach. • Various kinds of parameters like cost, response time, data processing time, and throughput are used.
An Optimized Task Scheduling Algorithm in Cloud Computing [38]	2016	LBM7	• Proposed an optimized task scheduling algorithm that takes advantage of different other available algorithms based on the situation. • It also considers features of cloud resources such as distribution and scalability.
Effective Load Balance Scheduling Schemes for Heterogeneous Distributed System [39]	2017	LBM 8	• Introduced load balancing algorithm for HeDs (Heterogeneous distributed system) focused on the minimization of load imbalance Factor (LIF). • This algorithm used an optimization method and then applied on FCC (folded cross cube) network. • Makespan average resource utilization and speedup are considered performance metrics.
An Adaptive Firefly Algorithm for Load Balancing in Cloud Computing [40]	2017	LBM 9	• Perform VM scheduling of data centers • Compared the result with the ACO algorithm in the context of load balancing.

Table 2 (continued)

Name of article	Year	Designated by	Brief Description
A PSO-based task scheduling algorithm improved using a load balancing technique for cloud computing environment [41]	2017	LBM 10	• Proposed static task scheduling technique using PSO. • Tasks are taken to be non-pre-emptive discipline and independent. • Improved performance of fundamental PSO algorithm.
Load balancing-based task scheduling with ACO in cloud computing [42]	2017	LBM 11	• Offered a meta-heuristic technique of ACO for the solution of task scheduling problems. • Focused on the minimization of makespan or computation time and achieving better balancing of load.
Load Balancing in Cloud Computing: A Big picture [43]	2018	LBM 12	• Discuss different load balancing strategies in various cloud Platform
Improved Effective Load Balancing Technique for Cloud [44]	2018	LBM 13	• In this paper proposed algorithm focused on requests with higher priorities gets higher importance. • The SJF algorithm and weighted RR algorithm collaborate and propose a new algorithm to handle various user requests.
Energy-Efficient Scheduling for Multiple Workflows in Cloud Environment [45]	2018	LBM 14	• Proposed scheduling heuristic for numerous tasks which are running onto cloud platform in a parallel manner. • Reduce energy consumption.
An Approach for Load Balancing in Cloud Computing Using JAYA Algorithm [46]	2019	LBM 15	• Offered an algorithm to minimize response time & service requests of data center and also improves the overall system performance. • Provide better-optimized results.
A Comparative Study on Load Balancing Algorithms in Software Defined Networking [32]	2019	LBM 16	• Introduce various SDN load balancing algorithms and analyze them to attain the loading result. • For making the entire network programmable, SDN (software-defined networking) technology introduced that separate data plane and control plane.
On the use of the genetic programming for balanced load distribution in software-defined networks [34]	2019	LBM 17	• Proposed GPLB (Genetic programming based load balancing). • GP deliberate load of integrated path based on the information collection from SDN controller.
Efficient load balancing techniques for multi-datacentre cloud milieu [47]	2020	LBM 18	• Introduced two multi-datacentre two-phase load adjustment approaches for apposite task scheduling and load balancing. • Provide significant improvement in makespan and resource utilization.
A Binary bird swarm optimization-based load balancing algorithm for cloud computing environment [48]	2021	LBM 19	• Offered an algorithm to increase the overall system performance by keeping the whole system in a balanced mode and reducing response time. • Reduce makespan and enhanced resource utilization.
A genetic load balancing algorithm to improve the QoS metrics for software defined networking for multimedia applications [37]	2022	LBM 20	• This paper considers the effect of the use of SDN services in different multimedia applications. • A GLBA algorithm is proposed and provide significant result based on different parameter like response time, throughput etc.

a network emulator that runs a cluster of switches, end hosts, links, and, routers on a single Linux kernel. It allows various profiling tools including *iperf* and *perf* [54].

f. **Java**: Various categories of users used and learned java other than languages. As compared to C++java have no limitation as the language for simulation. The most important characteristic of java is that it is lack a pointer and this allows more optimization possible for parallel and sequential codes both [54, 55]. C language also can be used for the coding of blocks while simulation because it is known as a mother of all programming languages and also suitable for traditional algorithm (See Table 3).

4.3 Simulation Environment

The computing environment gives information about the computer system, host (server), or workstation and the type of operating system, application, and peripherals they used. In cloud computing the computing environment would be either homogeneous or heterogeneous and are as follows:

a. **Homogeneous Environment**: This environment [56] refers to the H/W of the processor or software module (operating system, compiler) that uses the same storage representation and the same result for various operations performed on floating point numbers when the floating point number communicates with the processor. So the communication layer assures the exact transmission of the floating point value. It is a low-power consumption platform.

Table 3 Critical analysis based on simulator used

Algorithm	Cloud Sim	Cloud Analyst	Grid Sim	MATLAB	Mininet	Java 7	C language
LBM1	✓	x	x	x	x	x	x
LBM 2	✓	x	x	x	x	x	x
LBM 3	x	✓	x	x	x	x	x
LBM 4	✓	x	x	x	x	x	x
LBM5	✓	x	x	x	x	x	x
LBM 6	x	✓	x	x	x	x	x
LBM 7	✓	x	x	x	x	✓	x
LBM8	x	x	x	x	x	x	✓
LBM9	x	✓	x	x	x	x	x
LBM10	✓	x	x	x	x	x	x
LBM11	✓	x	x	x	x	x	x
LBM12	✓	x	x	x	x	x	x
LBM 13	✓	x	x	x	x	x	x
LBM 14	✓	x	x	x	x	x	x
LBM 15	x	✓	x	x	x	x	x
LBM 16	x	x	x	x	✓	x	x
LBM 17	x	x	x	x	✓	x	x
LBM 18	x	x	x	✓	x	x	x
LBM 19	✓	x	x	x	x	x	x
LBM 20	x	x	x	x	✓	x	x

Table 4 Critical analysis based on environment used

Algorithm	Environment	
	Homogeneous	Heterogeneous
LBM1	x	✓
LBM 2	✓	x
LBM 3	✓	x
LBM 4	x	✓
LBM 5	x	✓
LBM6	✓	x
LBM7	x	✓
LBM8	x	✓
LBM9	✓	x
LBM10	x	✓
LBM 11	x	✓
LBM 12	x	✓
LBM 13	x	✓
LBM 14	✓	x
LBM15	x	✓
LBM 16	x	✓
LBM 17	✓	x
LBM 18	x	✓
LBM 19	✓	x
LBM 20	✓	x

b. **Heterogeneous Environment**: In this environment [56], different sets of the architecture of the processor is used, and also used different storage representations. The result may differ from one processor to another based on floating point precision. So the communication layer doesn't assure exact transmission. It is a high-power consumption platform.

4.4 Comparison of Load Balancing Methods Based on QoS Parameters

QoS parameters are the attributes of the load balancing method that provide the efficiency of algorithms. Here Table 4 depicts the performance matrices based on Table 5.

5 Conclusion

Numerous algorithms have been suggested for the solution to the LB problem. This critical and comprehensive review provided good scope to researchers for the advancement of LB algorithms for the CCE. This paper will be helpful for the identification of research problems to further change different QoS parameters. This paper presented various LB techniques in different environments, simulators & tools, and OoS parameters i.e. waiting time, response time, throughput, reliability, energy consumption, etc. These parameters are crucial for efficient LB algorithms and play a vital role while selecting and designing new LB problem algorithms for further extension in future work.

Table 5 Critical analysis based on performance metrics of load balancing Algorithm

Algorithm/ Performance matrices	LBM 1	LBM 2	LBM 3	LBM 4	LBM 5	LBM 6	LBM 7	LBM 8	LBM 9	LBM 10	LBM 11	LBM 12	LBM 13	LBM 14	LBM 15	LBM 16	LBM 17	LBM 18	LBM 19	LBM 20
T_n	x	x	✓	x	x	✓	x	x	x	x	x	x	x	x	x	✓	✓	x	x	✓
T_s	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
R_e	x	x	x	x	x	x	x	x	x	x	✓	x	x	x	x	x	x	x	x	x
A_{cc}	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
P_b	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
M_s	x	x	✓	✓	x	✓	✓	✓	x	✓	✓	✓	✓	✓	x	x	x	✓	✓	x
S_c	x	x	✓	x	x	x	✓	x	x	x	x	x	x	x	x	x	✓	x	x	x
F_t	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
C_a	x	x	x	x	x	x	x	x	x	x	x	x	✓	x	x	x	x	x	x	x
MIG_{cost}	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
R_n	✓	✓	x	x	x	✓	x	x	✓	✓	x	x	x	x	✓	✓	✓	x	✓	✓
C_a	✓	✓	✓	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x	x
E_c	x	x	x	x	✓	x	x	x	x	x	x	✓	x	✓	✓	x	x	x	x	x
S_{speed}	x	x	x	x	x	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x

Funding Not Applicable.

Data Availability Availability at [14–23].

Code Availability http://www.cloudbus.org/gridsim/.

Declarations

Competing Interests The authors declare no competing interests.

References

1. Ekanayake, J., & Fox G. (2009). High performance parallel computing with clouds and cloud technologies. *International Conference on Cloud Computing*. Springer, Heidelberg.
2. Rajak, N., & Shukla, D. (2019). Comparative study of cloud computing and mobile cloud computing. *International Journal Of Engineering Sciences & Research Technology*, *7*(3).
3. Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing*. NIST Special Publication 800–145.
4. Pachghare, V. K. (2016) *Cloud computing*. PHI.
5. De, D. (2016). *Mobile Cloud Computing: Architecture, Algorithms and Applications*. CRC Press, Taylor and Francis Group.
6. Darji, V., Shah, J., & Mehta, R. (2014). Survey paper on various load balancing algorithms in cloud computing. *International Journal of Scientific & Engineering Research*, *5*(5), 583–588.
7. He, J. (2022). Cloud computing load balancing mechanism taking into account load balancing ant colony optimization algorithm. *Computational Intelligence and Neuroscience* 2022.
8. Alkhatib, Ahmad, A. A., et al. (2021). Load balancing techniques in cloud computing: Extensive review. *Advances in Science Technology and Engineering Systems Journal*, *6*(2), 860–870.
9. Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *SoftwPractExp*, *41*(1), 23–50.
10. Sankeerthi, S., & Sudha, K. (2018). Effective analysis of load balancing algorithms in cloud. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, *3*(1), 1436–1442.
11. Simaiya, D., & Paul, R. K. (2018). Review of various performcae evaluation issues and efficient load balancing for cloud computing. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, *3*(3), 943–951.
12. Choudhary, A., & Rajak, R. (2024). A novel strategy for deterministic workflow scheduling with load balancing using modified min-min heuristic in cloud computing environment. *Cluster Computing*. https://doi.org/10.1007/s10586-024-04307-8
13. Mishra, S. K., et al. (2018). Load balancing in cloud computing: A big picture. *Journal of King Saud University Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2018.01.003.
14. Haidri, R. A., Katti, C. P., Saxena, P. C. (2021). Capacity based deadline aware dynamic load balancing (CPDALB) model in cloud computing environment. *International Journal of Computers and Applications*, *43*(10), 987–1001.
15. Haidri, R. A., et al. (2022). A deadline aware load balancing strategy for cloud computing. *Concurrency and Computation: Practice and Experience*, *34*(1), e6496.
16. Rajak, N., Rajak, R., & Prakash, S. (2022). A workflow scheduling method for cloud computing platform. *Wireless Personal Communication*, *126*, 3625–3647. https://doi.org/10.1007/s11277-022-09882-w
17. Braun, T. D., Siegel, H. J., Beck, N., et al. (2001). A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *J Parallel DistribComput*, *61*, 810–837.
18. Pop, F., Dobre, C., & Cristea, V. (2009). Genetic algorithm for DAG scheduling in grid environments. In: *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*. pp 299–305.
19. Canon, L. C., & Jeannot, E. (2009). Evaluation and optimization of the robustness of DAG schedules in heterogeneous environments. *IEEE Transactions on Parallel and Distributed Systems*, *21*(4), 532–546.

20. Darbha, S., Dharma, P., & Agrawal (1994). SDBS: A task duplication based optimal scheduling algorithm. *Proceedings of IEEE Scalable High-Performance Computing Conference*. IEEE.
21. Sinnen, O. (2007). *Task scheduling for parallel systems*. Wiley.
22. Kumar, M. S., Gupta, I., & Jana, P. K. (2017). Delay-based workflow scheduling for cost optimization in heterogeneous cloud system. *2017 Tenth International Conference on Contemporary Computing (IC3)*. IEEE.
23. Gupta, I., Kumar, M. S., & Jana P. K. (2018). Efficient workflow scheduling algorithm for cloud computing system: A dynamic priority-based approach. *Arabian Journal for Science and Engineering*, *43*(12), 7945–7960.
24. Rajak, R., Kumar, S., Prakash, S., Rajak, N., & Dixit, P. (2023). A novel technique to optimize quality of service for directed acyclic graph (DAG) scheduling in cloud computing environment using heuristic approach. *The Journal of Supercomputing*, *79*, 1956–1979. https://doi.org/10.1007/s11227-022-04729-4
25. Babbar, H., Rani, S., Masud, M., Verma, S., Anand, D., & Jhanjhi, N. (2021). Load balancing algorithm for migrating switches in software-defined vehicular networks. *CMC-Comput Mater Continua*, *67*(1), 1301–1316.
26. Lu, Y., Xie, Q., Kliot, G., Geller, A., Larus, J. R., & Greenberg, A. (2011). Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services. *Perform Eval*, *68*(11), 1056–1071.
27. Hwang, K. (2005). *Advanced Computer Architecture: Parallelism, Scalability, Programmability*. 5th reprint. New Delhi:TMH Publishing Company, pp. 51–104.
28. Ramezani, F., Lu, J., & Hussain, F. K. (2014). Task-based system load balancing in cloud computing using particle swarm optimization. *International Journal of Parallel Programming*, *42*(5), 739–754.
29. Dam, S., Mandal, G., Dasgupta, K., & Dutta, P. (2015). Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing. *Proceedings of the 2015 Third International Conference on Computer Communication Control and Information Technology (C3IT)*, 1–7. https://doi.org/10.1109/C3IT.2015.7060176.
30. Pan, K., & Chen, J. (2015). Load balancing in cloud computing environment based on an improved particle swarm optimization. In: *6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, pp 595–598.
31. Yakhchi, M., Ghafari, S. M., Yakhchi, S., Fazeli, M., & Patooghi, A. (2015). Proposing a load balancing method based on Cuckoo Optimization Algorithm for energy management in cloud computing infrastructures. In *2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO)*. pp. 1–5, IEEE, May.
32. Zhu, Y., Zhao, D., Wang, W., & He, H. (2016). A novel load balancing algorithm based on improved particle swarm optimization in cloud computing environment. In: *International Conference on Human-Centered Computing*. Springer, pp. 634–645.
33. Mittal, S., & Katal, A. (2016). An optimized task scheduling algorithm in cloud computing. In *IEEE 6th International Conference on Advanced Computing (IACC)*. (pp. 197–202). IEEE. https://doi.org/10.1109/IACC.2016.45.
34. Khan, Z., Alam, M., & Haidri, R. A. (2017). Effective Load balancing in cloud computing. *International Journal of Electrical & Computer Engineering7*(5), 2757–2765.
35. Kaur, G., & Kaur, K. (2017). An adaptive firefly algorithm for load balancing in cloud computing. In *Proceedings of Sixth International Conference on Soft Computing for Problem Solving. Advances in Intelligent Systems and Computing*, vol 546. Springer. https://doi.org/10.1007/978-981-10-3322-3_7.
36. Ebadifard, F., & Babamir, S. M. (2018). A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment. *Concurrency and Computation: Practice and Experience*, 30(12), e4368.
37. Gupta, A., & Garg, R. (2017). Load balancing based task scheduling with ACO in cloud computing. In *2017 International Conference on Computer and Applications (ICCA)*, pp. 174–179, IEEE.
38. Mishra, S. K., Sahoo, B., Parida, P. P. (2020). Load balancing in cloud computing: A big picture. *Journal of King Saud University - Computer and Information Sciences*, *32*(2), 149–158.
39. Kulkarni, S. V., & Kodli, S. B. (2018). Improved effective load balancing technique for cloud. *International Journal of Scientific Research in Science and Technology*, *4*(9), 368–375.
40. Ritu, G., & Shukla, N. (2018). Energy efficient scheduling for multiple workflows in Cloud Environment. *International Journal of Information Technology and Web Engineering*, *13*, 14–34. https://doi.org/10.4018/IJITWE.2018070102.
41. Mohanty, S., et al. (2019). An approach for load balancing in cloud computing using JAYA algorithm. *International Journal of Information Technology and Web Engineering (IJITWE)*, *14*(1), 27–41.

42. Joshi, N., & Gupta, D. (2019). A comparative study on load balancing algorithms in software defined networking. In N. Kumar & R. Venkatesha Prasad (Eds.), *Ubiquitous communications and network computing. UBICNET 2019*. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (Vol. 276). Cham: Springer. https://doi.org/10.1007/978-3-030-20615-4_11
43. Jamali, S., Badirzadeh, A., Siapoush, M. S. (2019). On the use of the genetic programming for balanced load distribution in software-defined networks. *Digital Communications and Networks*, *5*(4), 288–296. https://doi.org/10.1016/j.dcan.2019.10.002.
44. Sharma, S.C.M., Rath, A.K. & Parida, B.R. (2022). Efficient load balancing techniques for multi-data-center cloud milieu. *International Journal of Information Technology*, *14*, 979–989.
45. Mishra, K., & Majhi, S. K. (2021). A binary bird swarm optimization based load balancing algorithm for cloud computing environment. *Open Computer Science*, *11*(1), 146–160.
46. Babbar, H., Parthiban, S., Radhakrishnan, G., & Rani, S. (2022). A genetic load balancing algorithm to improve the QoS metrics for software defined networking for multimedia applications. *Multimedia Tools and Applications*, *81*, 9111–9129. https://doi.org/10.1007/s11042-021-11467-x.
47. Calheiros, R. N. (2009). Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services. arXiv preprint arXiv:0903.2525.
48. Wickremasinghe, B., Calheiros, R. N., & Buyya, R. (2010). CloudAnalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In *433-659 Distributed Computing Project*. University of Melbourne.
49. Patel, H., & Patel, R. (2015). Cloud analyst: An insight of service broker policy. *International Journal of Advanced Research in Computer and Communication Engineering*, *4*(1), 122–127.
50. Simatos, C. (2002). *Making simjava count. MSc. Project report*. The University of Edinburgh.
51. Yazdi, N. T., & Yong, C. H. (2015). Simulation of Multi-agent Approach in Multi-cloud Environment using Matlab. *2015 Seventh International Conference on Computational Intelligence Modelling and Simulation (CIMSim)*, 77–79. https://doi.org/10.1109/CIMSim.2015.22.
52. Team, M. (2012). Mininet: An instant virtual network on your laptop (or other pc).
53. Alshammari, D., Singer, J., & Storer, T. (2018). Performance evaluation of cloud computing simulation tools. *IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, pp. 522–526, https://doi.org/10.1109/ICCCBDA.2018.8386571.
54. Fox, G., & Furmanski, W. (1997). Java and Web Technologies for Simulation and Modelling in Computational Science and Engineering.
55. Esquembre, F. (2004). Easy Java simulations: A software tool to create scientific simulations in Java. *Computer Physics Communications*, *156*, 199–204. https://doi.org/10.1016/S0010-4655(03)00440-5.
56. Mittal, S., & Suri, P. K. (2012). A comparative study of various Computing Processing environments: A review. *International Journal of Computer Science and Information Technologies (IJCSIT)*, *3*, 5215–5218.

Ms. Anjali Choudhary currently pursuing her Ph.D. in Computer Science from the Department of Computer Science & Application, Dr. HarisinghGour, Central University, Sagar, Madhya Pradesh, India. She completed her B.Sc. (CS) & M.Sc. (CS) in 2013 & 2015 from RDVV University, Jabalpur, Madhya Pradesh. She has also awarded by Gold Medal in M.Sc. 2015. Her area of interest is cloud computing and IoT. She has published several papers in journals, conferences& book chapters.

Dr. Ranjit Rajak received his MCA, M.Tech. and Ph.D. degree in Computer Science and Technology from JNU, New Delhi, India in the year 2007, 2009 and 2014. His area of interests includes Parallel Computing, Scheduling Problem, WSN and Cloud Computing.He is working as an Assistant Professor since May, 2013 in the Department of Computer Science and Applications, Dr.Harisingh Gour Central University, Sagar (M.P) and has published over 29 papers in the journals, conferences and book chapters of national and international repute. He is also a member of IAENG and CSI.

Dr. Shiv Prakash received his Ph.D. degree and M.Tech. degree in Computer Science and Technology from Jawaharlal Nehru University, New Delhi in 2014 and 2010 respectively. His current research interest focuses on the Internet of Connected Vehicles (IoV), Electronic Vehicles (EV), Artificial Intelligence, Big Data Analytics, Biometric Security, Cloud Computing, Computer Networks, Machine Learning, Network Security, and IoT use cases of Sensor Networks. He has an excellent educational record throughout. He is a Reviewer of several referred International journals of repute (including ACM, IEEE, Taylor and Francis, Elsevier, Springer, Wiley, etc.)and published 40+research papers in various peer-reviewed International Journals and Transactions (including IEEE, Taylor and Francis, Elsevier, Springer, Wiley, American Scientific Publishers, etc.) and around 20+research papers in proceedings of various peer-reviewed conferences in India and abroad. He has published several papers in journals having very high ISI impact factors. He is a member of various International Associations, Societies, and Scientific Committees. He is engaged in research in the collaboration with National and International organizations of repute.