Issues and Challenges of Load Balancing Techniques in Cloud Computing: A Survey

PAWAN KUMAR and RAKESH KUMAR, NITTTR, Chandigarh

With the growth in computing technologies, cloud computing has added a new paradigm to user services that allows accessing Information Technology services on the basis of pay-per-use at any time and any location. Owing to flexibility in cloud services, numerous organizations are shifting their business to the cloud and service providers are establishing more data centers to provide services to users. However, it is essential to provide cost-effective execution of tasks and proper utilization of resources. Several techniques have been reported in the literature to improve performance and resource use based on load balancing, task scheduling, resource management, quality of service, and workload management. Load balancing in the cloud allows data centers to avoid overloading/underloading in virtual machines, which itself is a challenge in the field of cloud computing. Therefore, it becomes a necessity for developers and researchers to design and implement a suitable load balancer for parallel and distributed cloud environments. This survey presents a state-of-the-art review of issues and challenges associated with existing load-balancing techniques for researchers to develop more effective algorithms.

CCS Concepts: • **Computing methodologies** → *Distributed algorithms*; *Concurrent algorithms*;

Additional Key Words and Phrases: Load balancing, cloud computing, resource allocation, task scheduling, virtual machine, workload management, optimization

ACM Reference format:
Pawan Kumar and Rakesh Kumar. 2019. Issues and Challenges of Load Balancing Techniques in Cloud Computing: A Survey. *ACM Comput. Surv.* 51, 6, Article 120 (February 2019), 35 pages.
https://doi.org/10.1145/3281010

1 INTRODUCTION

With the rapid growth in Information Technology (IT), cloud computing has emerged as the substitute for traditional computing technologies for providing services to clients at any time and any location on a pay-per-use basis (Brown 2017; Buyya et al. 2010), which enables users to access a pool of configurable computing resources (servers, storage, networks, applications). Several companies (e.g., Amazon Web Services (AWS), Microsoft Azure, Google, IBM cloud, Rackspace, Red Hat, Verizon cloud, VMware) (Technavio 2018), also called cloud computing providers, offer these cloud services to users. The primary aim of the cloud is to use the distributed resources

Authors' addresses: P. Kumar and R. Kumar, Department of Computer Science and Engineering, National Institute of Technical Teacher's Training and Research, Sector-26, Chandigarh, India; emails: pawan.cse@nitttrchd.ac.in, raakeshdhiman@gmail.com.

0360-0300/2019/02-ART120 $15.00
https://doi.org/10.1145/3281010

effectively in order to attain high throughput and performance. It enables the cloud to resolve the problems that require high computing power. It also allows distribution of the resources all around the world to execute the tasks at different data centers to provide faster services to clients (Dasgupta et al. 2013). Along with cloud computing distribution systems (Milani and Navimipour 2016; Navimipour 2015), many other distribution systems are available, such as grid computing (Khanli et al. 2008) and peer-to-peer computing (Navimipour and Milani 2015), which allow resource sharing and data transfer facilities. This provides business opportunities to both cloud service providers to establish new data centers and cloud users to put their business on the cost-efficient cloud. Cloud computing allows users to scale in or out virtual resources dynamically without human interaction according to their computing requirements (Chiregi and Navimipour 2016).

Cloud computing can be categorized in two ways: based on location or services offered. Based on location, a cloud can be public, private, hybrid, or community. Public cloud services are available to anyone and the infrastructure is located on the premises of a service-providing company. Public clouds are most vulnerable to various attacks, but they are most cost-effective. A private cloud is exclusively available to a specific user or organization. It provides the highest security and control level to the user with higher cost. A hybrid cloud is a combination of public and private cloud that is used for different purposes based on organizational requirements. A community cloud consists of a common infrastructure used by many organizations that have shared data and management. Based on services, the cloud is mainly classified as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), or Software as a Service (SaaS). In IaaS, the cloud offers elementary IT resources such as networking features, computers, more flexibility, and control over the computing resources. PaaS removes the need for organizations to handle basic infrastructure—typically, computer hardware and operating system—(OS) and allows you to focus on the deployment of applications. SaaS allows users to concentrate on use of particular software rather than thinking about how the infrastructure and services are managed. Along with these services, cloud computing provides Database as a Service (DaaS) (Hacigumus et al. 2002), Expert as a service (EaaS) (Jafari et al. 2015), Storage as a Service (SaaS), Network as a Service (NaaS), Security as a Service (SECaaS) (Candrlic 2013), Communication as a Service (CaaS), Monitoring as a Service (MaaS), and Testing-as-a-service (TaaS) (Dhillon 2015), which are accessed by the user for different application purposes. Numerous cloud applications are available on different aspects of education, health monitoring (Punj and Kumar 2018), data analysis, and robotics. Users can access them without knowledge of technology required for their computing environment. According to Banerjee et al. (2015), the cloud also provides the flexibility and scalability to release and acquires the various configurable resources depending on application requirement.

As the cloud provides different services, it needs quality of service (QoS) monitoring to assess the offered services for fulfilling user demands and to maintain the service level agreement (SLA). During this process, the cloud may pose several issues and challenges, such as load balancing (Chen et al. 2017; Mohamed et al. 2013; Nuaimi et al. 2012; Pacini et al. 2015; Suresh et al. 2014), performance analysis and modeling (Garg et al. 2013; Ghosh et al. 2013; Li et al. 2012; Mauch et al. 2013; Miguel et al. 2015; Sousa et al. 2015), throughput and response time (Pacini et al. 2015), security and privacy issues (Khan et al. 2013; Khorshed et al. 2012; Lombardi and Di Pietro 2011; Subashini and Kavitha 2011; Zissis and Lekkas 2012), resource management (Jennings and Stadler 2015; Marinescu et al. 2017), and QoS. Currently, load balancing in the cloud (LBC) is one of the main challenges that allows avoiding the situation of overloading/underloading in virtual machines during task computation. Thus, there is a need to identify the issues that affect LBC and develop an effective load balancing technique for cloud environments.

The main objectives of this study are as follows:

- To study various load balancing techniques existing in the literature.
- To classify various load balancing techniques and overview the existing challenges and issues in load balancing.
- To outline the research areas to improve the load balancing techniques in future.

1.1 Need of Load Balancing

Load balancing provides the facility to distribute the workload equally on available resources. Its objective is to provide continuous service in case of failure of any service's component by provisioning and deprovisioning the application instances along with proper utilization of resources. In addition, load balancing aims to minimize the response time for tasks and improve resource utilization, which enhances system performance at lower cost. Load balancing also aims to provide scalability and flexibility for those applications whose size may increase in future and requires more resources as well as to provide priority to jobs that need instant execution as compared with another jobs. Other objectives of load balancing are reducing energy consumption and carbon emission, avoiding bottlenecks, resource provisioning, and fulfilling QoS requirements for improving load balancing. There is a need for proper workload mapping and load balancing techniques that consider different metrics.

1.2 Motivation

- Load balancing in the cloud is the process of equally distributing the workload on virtual machines for proper utilization of resources. In this survey, various load-balancing algorithms are accentuated based on different metrics.
- Load balancer helps in allocation of resources to the tasks fairly for resource utilization and user satisfaction at minimum cost, which motivates us to find issues in load balancing and to work on resolving them.
- Based on continuous demand and increasing workload in the field of cloud computing, we have recognized the need of load balancing among cloud resources. Therefore, on the basis of available research, the existing work has been identified and summarized in a systematic way that depicts issues and challenges for future research work.

1.3 Article Organization

This article is organized as follows: Section 2 discusses surveys done earlier on load balancing in the cloud. Section 3 describes the survey technique adopted to find available research papers, search criteria, the source of information, and quality assessment. Section 4 presents various challenges in load balancing. Section 5 discusses classification, metrics, and policies of load balancing. Section 6 contains the literature review of various load-balancing techniques and some simulation tools used to demonstrate these load-balancing and resource-scheduling tools. In Section 7, we present a discussion about the various mentioned techniques. Section 8 describes the issues and challenges faced by existing load-balancing techniques. We present our concluding remarks and state the future trends in Section 9.

2 RELATED SURVEYS

Global research communities are showing interest in designing and developing optimum resource utilization techniques as current research and reviews are drawing their attention to it. Load balancing in the cloud is a process of distributing the execution load on uncommitted virtual machines to increase system throughput. Minimization of response time and execution cost (Goyal and Verma 2016) is needed to increase the success of cloud-computing environments. Along with load balancing, there are various challenges, such as resource scheduling, performance monitoring, QoS management, energy consumption, and service availability in the cloud (Kaur and Luthra 2012; Malladi 2015). Numerous research efforts have been made in cloud computing, load balancing, security, energy consumption, resource management, and the like. Load balancing and task scheduling are main concerns and need more research. We provide a comprehensive review of various types of scheduling, load balancing, and task-based load-balancing techniques of the cloud in this article. In this section, we review some articles that have significantly surveyed load-balancing techniques.

An extensive review on load balancing in the cloud was conducted by Ghomi et al. (2017), who split the review into seven different categories: Hadoop MapReduce load balancing techniques, Natural phenomena-based load balancing techniques, Agent-based load balancing techniques, General load balancing techniques, Application oriented load balancing techniques, Network-aware task scheduling and load balancing, Workflow specific scheduling algorithms They discussed various issues and challenges in existing load-balancing techniques along with future directions. They have concentrated mainly on Hadoop MapReduce and energy efficiency, which concern cloud load balancing; still, their work lacks in task-based and cluster-based load balancing, which are also important issues for executing the tasks on available resources. Thus, there is a need to identify the issues in task-based and cluster-based load balancing in today's highly scalable and largely distributed computing environment.

Milani and Jafari (2016) reviewed various existing load-balancing schemes and classified them into dynamic and hybrid subdomains. They described the behavior of these techniques based on different parameters, along with their advantages, disadvantages, and challenges. They stated the issues with these algorithms to develop more efficient algorithms for minimizing resource consumption and power consumption and to make load-balancing techniques more effective. However, they did not discuss task-based load balancing, cluster-based load balancing, and energy consumption issues.

Singh et al. (2017) presented an extensive review on meta-heuristic—based task scheduling algorithms, whereas the review in Kalra and Singh (2015) was limited to meta-heuristic techniques for work-flow scheduling only. The authors Singh et al. (2017) had identified various issues related to task scheduling in the cloud and presented a comparative analysis for both dependent and independent tasks based on the meta-heuristic approach. In Singh et al. (2017) and Kalra and Singh (2015) most of the literature concentrates on Genetic Algorithms, Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) based scheduling techniques, and lacks coverage of task-based load balancing, which is also currently a primary issue in the cloud. Singh et al. (2017) also have not assessed other major issues such as carbon emission, cluster-based load balancing, and minimum resource consumption.

A review of automatic cloud resource management was conducted by Singh and Chana (2015), who presented the study based on six different perceptions—resource allocation, workload scheduling, monitoring, QoS requirement, design of application, self-management and discussed taxonomy based on different characteristics. They have concentrated on improving management and use of resources, while task execution time and response time are more important to provide QoS along with fulfilling the SLA.

A comparative analysis of most-used load-balancing techniques based on metrics (such as response time, migration time, scalability, and resource use) in a distributed environment, cluster system in distributed cloud computing environment is presented in Ivanisenko and Radivilova (2015). The authors have presented the features, advantages, and disadvantages of most-used load-balancing algorithms but have missed the issues in existing techniques, challenges, and future trends.

An analysis of load-balancing algorithms based on user requirements specified in SLA for various cloud environments is presented by Katyal and Mishra (2014). The authors review these techniques based on different categories, including task dependencies, geographical distribution of nodes, and cloud environment. They discuss advantages, disadvantages, and challenges in existing techniques in the main categories but lack an evaluation based on various load-balancing parameters.

Based on the analysis, we observed that there is no comprehensive review about task scheduling and load-balancing techniques that presents the importance of these techniques with categorization and future issues. In this article, we developed a list of questions to select the most important research papers for review and categorized them to answer these questions.

3 SURVEY TECHNIQUE

In this article, we have followed various guidelines in different fields presented by Kitchenham (2004), Kitchenham et al. (2009), Kupiainen et al. (2015), Charband and Navimipour (2016), and Navimipour and Charband (2016) for undertaking a methodical survey and focused on cloud load-balancing related research. We formulated the review method, investigating the method, taking the results and exploring the challenges. In this section, we discuss the source of information for research articles, selection criteria, quality assessment, and evaluation of results. Table 1 contains our list of research questions to plan the survey on load balancing and to determine the current issues pertaining to load balancing. Figure 1 shows the seven-stage process used to search for and identify articles based on inclusion-exclusion criteria.

3.1 Source of Information

We broadly searched for journal and conference research articles in Scopus, Web of Science, Google Scholar, books, and magazines as a source of data to extract relevant articles. The following databases has been used in our search:

- IEEE Xplore (http://ieeexplore.ieee.org)
- Springer (https://link.springer.com)
- ScienceDirect (http://www.sciencedirect.com)
- Google Scholar (https://scholar.google.co.in)
- Scopus (https://www.scopus.com)
- ACM Digital Library (https://www.acm.org/digital-library)
- Taylor & Francis (https://www.taylorandfrancis.com)

In Figure 2, we show the percentage of papers reviewed from different sources.

3.2 Search Criteria

We defined the keywords to search in the abovementioned databases. The keywords “Load Balancing” and “cloud” were involved in the abstract of every search. It is a common method and time-consuming. We searched for different synonyms and related keywords that matched our results, including “task migration,” “resource utilization,” “distributed,” “VM migration,” “workload,” and “cluster.” We refined the query again to match the specific results and applied again on Title, Abstract of the research paper. We conducted the study between January 2010 to January 2018.

Table 1. Review Questions

No.	Research Questions	Motivation
1.	What is the significance of load balancing in the cloud?	Mainly, the aim is to identify various load-balancing studies/articles published over time and their importance with increasing use of the cloud. Numerous load-balancing techniques have been proposed so far, which need to be evaluated based on various load-balancing metrics. This survey also aims to identify the issues and challenges in existing load-balancing techniques to ensure QoS-based services. Here, we discuss various load-balancing techniques based on different categories.
2.	Why is load balancing required in the cloud?	
3.	Do the existing techniques fulfill load-balancing essentials metrics?	
4.	What is the current status of cloud load balancing?	
5.	What are the criteria to select a suitable service provider for the consumer?	
6.	What are the criteria to negotiate between cloud service providers and cloud consumers?	
7.	How does one minimize power consumption in cloud data centers and its impact on environment?	
8.	What are the new algorithms that need to be proposed to improve system performance?	
9.	Which resources and parameters are most important in cloud performance?	
10.	How does one manage the resources to overcome the problem of resource overloading and underloading?	
11.	How does one reduce response time and improve resource use?	
12.	What are the QoS requirements that the user expects to use the cloud?	
13.	How does one develop load-balancing techniques that fulfill the user's QoS requirements?	Various research papers need to be identified from different load-balancing categories to reveal vital research problems. With increasing demand for the cloud over time, it becomes essential to define load-balancing criteria and to develop techniques that handle large numbers of user requests effectively. Various questions discussed here will help in identification of future research areas.
14.	How does one validate the existing and developing load- balancing techniques through available tools?	
15.	How does one develop an architecture that fulfills the user's essential QoS need?	
16.	How is the SLA defined? What are the criteria to define SLA violations rules?	
17.	Which simulation tools are available for validation of load-balancing techniques?	It is useful to identify various cloud simulation tools that will help algorithm developers to validate load-balancing and resource-scheduling techniques.
18.	Which algorithm validation parameters are considered by these simulators?	

From the searched papers, we included the papers in our survey that meet the Quality Assessment Check-list (QAC) based on Kitchenham et al. (2009). It comprisespapers from peer-reviewed journals, books, conferences, magazines, symposiums, white papers, and websites.

3.3 Quality Assessment

On searched articles, we applied a quality assessment process following inclusion and exclusion criteria as shown in Table 2. We discovered 1,024 articles based on basic keywords from different journals from various sources, including IEEE Xplore, Springer, Science Direct, and ACM. We excluded some articles based on titles that were not relevant to our study. After reading the abstract,

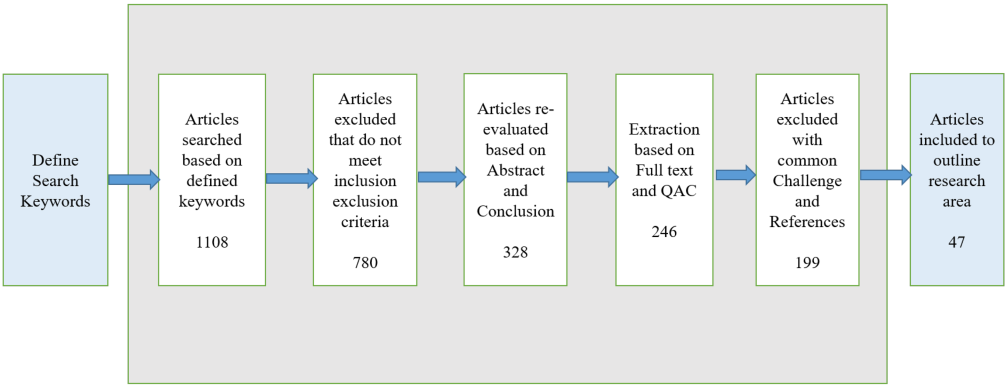
Define Search Keywords
Articles searched based on defined keywords 1108
Articles excluded that do not meet inclusion exclusion criteria 780
Articles re-evaluated based on Abstract and Conclusion 328
Extraction based on Full text and QAC 246
Articles excluded with common Challenge and References 199
Articles included to outline research area 47

Fig. 1. Article identification process overview.

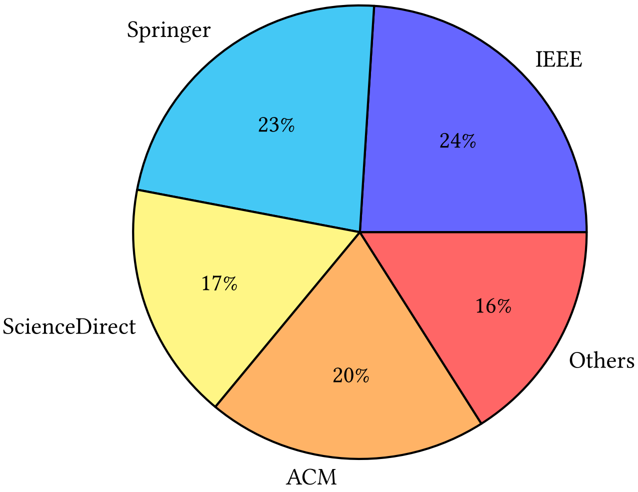

Fig. 2. Source of surveyed load balancing research papers.

we excluded some articles that did not meet our criteria. For the remaining papers, we reviewed the articles completely and found 47 research papers that are included in our review.

4 CHALLENGES IN LOAD BALANCING IN THE CLOUD

Cloud computing technology is becoming the target of more advanced research in the field of data and computation in terms of theoretical and practical aspects. However, cloud computing research is facing a lot of issues, load balancing being one of the prominent challenges that needs special attention. In addition, several other issues such as virtual machine (VM) migration, VM security, user QoS satisfaction, and resource use require equal attention in order to find the best possible solution for improving cloud resource use. A list of a few load-balancing issues are discussed below:

(1) **Geographical Distributed Nodes:** In general, data centers in the cloud are geographically distributed for computing purposes. In these centers, spatially distributed nodes are treated as a single location system for efficient execution of user requests. Some load-balancing techniques are designed for a smaller area where they do not consider the factors such as network delay, communication delay, distance between the distributed

Table 2. Paper Inclusion-Exclusion Criteria

Criteria	
Inclusion	1. Clearly describes load balancing technique from service provisioning prospective. 2. Published in cloud computing field. 3. Peer-reviewed and written in English language. 4. Published in reputable journals, conferences, and magazines. 5. Written by academic or industrial researchers.
Exclusion	1. Does not focus on load balancing in the cloud. 2. Has common challenges and references.

computing nodes, distance between user and resources, and so on. Nodes located at very distant locations are a challenge, as these algorithms are not suitable for this environment. Thus, designing load-balancing algorithms for distantly located nodes should be taken into account.

(2) **Single Point of Failure:** Various dynamic load-balancing algorithms are designed where some techniques are non-distributed and the decisions for load balancing are made by the central node. If the central device crashes, then it will affect the overall computing environment. Thus, there is a need to develop some distributed algorithms in which a single node does not control the whole computing system.

(3) **Virtual Machine Migration:** Virtualization allows creation of several VMs on a single physical machine. These VMs are independent in nature and have different configurations. If a physical machine gets overloaded, some VMs need to transfer to a distant location using a VM migration load-balancing approach.

(4) **Heterogeneous Nodes:** During early research in cloud load balancing, researchers theorized about homogeneous nodes. In cloud computing, user requirements change dynamicallywhich requires executing them on heterogeneous nodes for effective resource use and minimizing response time. Therefore, the invention of efficient load-balancing techniques for the heterogeneous environment is a challenge for researchers.

(5) **Storage Management:** Cloud storage has resolved the problem of older traditional storage systems that required personnel management and a high cost of hardware. The cloud allows users to store data heterogeneously without any access problems (Wu et al. 2012). Cloud storage is increasing day by day, which requires storing a replication of data for efficient access and consistency of data. Full data replication schemes are not very efficient due to duplicate data storage policy on replication points. Partial replication can be sufficient but there can be an issue of dataset availability and it increases the complexity of load-balancing techniques. So, an efficient load-balancing technique needs to be developed that considers the distribution of application and related data based on a partial replication system.

(6) **Load-Balancer Scalability :** On-demand availability and scalability of cloud services allow users to access services any time to scale down or scale up quickly. A good load balancer should consider quick changes in demands in terms of computing power, storage, system topology, and so on, to facilitate these changes efficiently (Ray and Sarkar 2012).

(7) **Algorithm Complexity:** In cloud computing, algorithms should be simple and easy to implement. A complex algorithm will reduce the performance and efficiency of the cloud system.

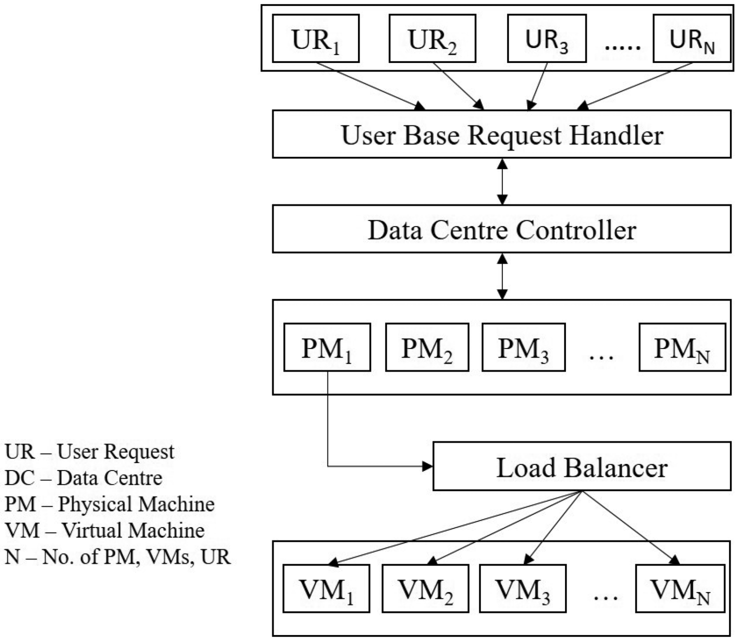

Fig. 3. Load balancing model.

5 LOAD BALANCING MODEL, CLASSIFICATION, METRICS AND POLICIES

The cloud provides on-demand access to a shared pool of resources (e.g., server, storage, and network (Zeng et al. 2015; Zhang et al. 2010)), which requires a great amount of control and management of the user's workload and resources. To manage user requests for the available resources, a good load balancer is needed to allocate the tasks to VMs based on their QoS requirements (Bhardwaj and RamaKrishna 2018). A model and workflow of the load balancer are shown in Figure 3. The cloud observes a high variation in user requests that require the dynamic environment to execute the tasks. When the cloud detects any request from the user base, a service broker on the cloud identifies the availability of resources and consults with other brokers about performance and cost of resources. After analyzing the available resources, the broker transfers the user request to the selected data center where the Data Center Controller (DCN) accepts them for further processing. The data center consists of physical resources/machines that take the requests and transfer them to the load balancer available on the server, which distributes the tasks to virtual machines (VM) to execute. During this process, the load balancer returns the availability of VMs from the state table and updates the state table after allocation. If no free VM is detected, the DCN puts the tasks in a queue and waits for resource availability. Once a VM finishes a task, the load balancer allocates it to another task for processing. The data center also has a VM manager that handles all of the VMs on physical servers. The load balancer is responsible for allocating a suitable VM to the tasks where task assignment is a critical issue in the cloud. The load balancer also ensures that VMs are not overloaded or underloaded. If some VMs are underloaded or free while others are overloaded, then the performance of the system and QoS will deteriorate, which may lead users to abandon the services. Virtualization technology helps to use the physical resources by sharing them among VMs. Hypervisors provide the facility of virtualization to create and manage the VMs in the cloud. A hypervisor also provides four important operations: provision, multiplexing, suspension, and live migration for load balancing (Hwang et al. 2013) of tasks and VMs. A

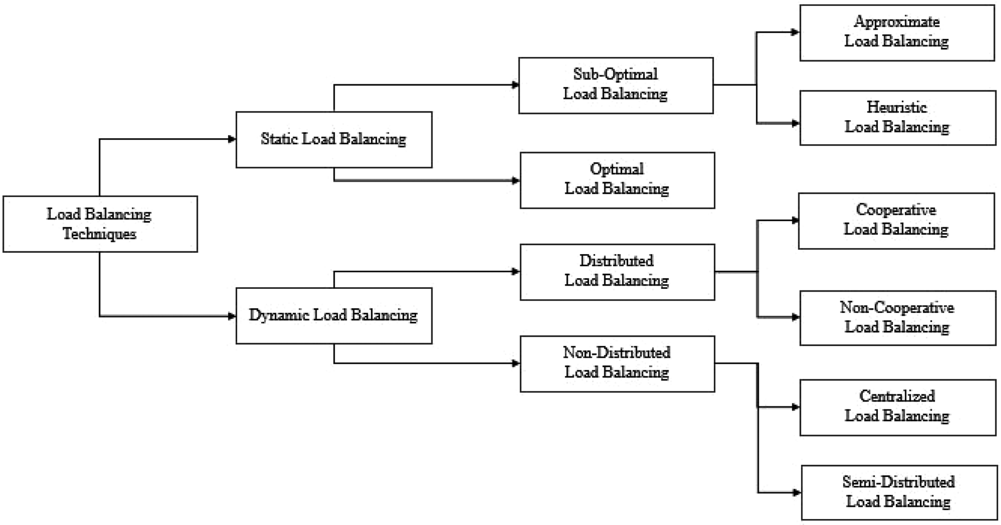
Load Balancing Techniques
Static Load Balancing
Dynamic Load Balancing
Sub-Optimal Load Balancing
Optimal Load Balancing
Distributed Load Balancing
Non-Distributed Load Balancing
Approximate Load Balancing
Heuristic Load Balancing
Cooperative Load Balancing
Non-Cooperative Load Balancing
Centralized Load Balancing
Semi-Distributed Load Balancing

Fig. 4. Classification of load-balancing strategies based on system state.

load balancer mainly considers scheduling the tasks and allocating them to a perfect resource. A good load balancer will increase resource use and availability and minimize the response time for tasks.

5.1 Classification of Load Balancing Strategies based on System State

Load balancing techniques are primarily classified based on the state of the system and process initiation. Load-balancing approaches based on system state are further classified into static and dynamic categories, as shown in Figure 4. Based on process initiation, they are classified as sender initiated, receiver initiated, and symmetric, as detailed below.

- **Sender Initiated:** In this technique, if a node is overloaded, it looks for other nodes that are lightly loaded to share the workload. The sender initiates the process to find the underloaded nodes when nodes get congested.
- **Receiver Initiated:** In this technique, receiver or lightly loaded nodes look for heavily loaded nodes to share the workload.
- **Symmetric:** In this technique, both sender-initiated process and receiver-initiated process techniques are combined.

Based on state of the system, load-balancing techniques can be divided into following subcategories:

- **Static:** Static load-balancing techniques follow a fixed set of rules that are not dependent on the system's current state. Static algorithms are not flexible and require prior knowledge of resources, such as communication time, memory and storage capacity of nodes, processing power of nodes, and so on. This technique is simple and easy but generally unable to detect the attached servers, which leads to uneven distribution of resources (Chen et al. 2017). The main issue with this technique is that the current state of the system is not considered during decision making. Thus, it is unsuitable for distributed systems that change state dynamically. Static techniques work very well only if lower load fluctuation occurs in the nodes. Static load-balancing techniques are categorized as follows:

  - **Optimal:** In optimal techniques, the DCN collects information about resources and sends tasks to the load balancer, which does an optimal allocation in minimum time.
  - **Suboptimal:** If the load balancer is not able to determine the optimal decision, it will calculate a suboptimal solution. A few static techniques are Min-Min, Max-Min, Round Robin, Shortest Job First, Two-phase Opportunistic Load Balancing (OLB), and Central Load Balancing for Virtual Machines.
- **Dynamic:** These techniques consider the current state of the system and make a decision on that basis. The main advantage of these techniques is that they allow transferring the tasks from an overloaded machine to an underloaded machine. Dynamic load balancing techniques are flexible, which leads to improvement in system performance. During processing, a dynamic technique takes the following steps. It continuously monitors the load of nodes. At a given time interval, it exchanges load and state information between nodes to calculate the nodes' workload and redistributes the workload between nodes. If a node gets overloaded, it transfers the load to an underloaded node. Some load-balancing techniques are Agent-based Load Balancing (Singh et al. 2015), Honey Bee Behavior Inspired Load Balancing (Babu and Krishna 2013), Ant Colony Optimization (Nishant et al. 2012), and Throttled (Domanal and Reddy 2013). Further dynamic load-balancing techniques can be categorized as follows:
  - **Distributed:** In distributed techniques, all nodes participate in load distribution, such as task scheduling or resource allocation (Cosenza et al. 2011; Shi et al. 2011). All nodes maintain an information base for communication to distribute and redistribute the tasks efficiently. Distributed algorithms can be in cooperative or non-cooperative form to interact with each other. If all nodes in the system work together to achieve a common goal or decision making, then it is called cooperative; otherwise, it is non-cooperative.
  - **Non-distributed:** In non-distributed techniques, a single node or some nodes make the decision for load distribution (Ahmad and Ghafoor 1991; Das et al. 2003). Non-distributed techniques can be centralized or semi-distributed in behavior. In centralized techniques, a single node performs all load distribution activities and is responsible for load balancing. Fault tolerance is an issue with centralized techniques and, in the case of single-node failure, node information can be lost and may not be recovered. In the semi-distributed technique, clusters are formed in nodes and each cluster works as a centralized technique.

5.2 Load-Balancing Metrics

To attain better resource use and improve performance, a load balancer becomes compulsory to disperse the computation load to available resources. Several load balancing techniques and various metrics to apply to these techniques are proposed by various researchers for higher user satisfaction and resource use. To increase overall system performance, researchers need to ensure that all parameters are fulfilled optimally. These metrics have been discussed by Chen et al. (2017), Daraghmi and Yuan (2015), Abdulhamid et al. (2014), Kansal and Chana (2012), Randles et al. (2010), Babu and Krishna (2013), Voorsluys et al. (2011), and Ramezani et al. (2014), as follows:

- **Performance:** System effectiveness must be validated after implementing the technique as compared with other existing techniques for load balancing.
- **Response Time:** The total time taken to complete a submitted request on the system.
- **Throughput:** The total amount of submitted tasks or processes completed in a unit of time on a system. The higher the throughput, the better the system performance.
- **Scalability:** The capability of the system to accomplish uniform load balancing when the required number of nodes increases.

- **Fault Tolerance:** The capability of the load-balancing technique to perform uniformly in the case of breakdown of any link or node.
- **Migration Time:** Migration time is used to figure out the total time taken to transmit a request/task to an underloaded machine from an overloaded machine. The lower the migration time, the better the performance of the cloud system.
- **Resource Use:** This is evaluated to ensure that all resources are properly used in the cloud system. Higher resource use will lead to minimizing overall cost and to reduced energy expenditure and carbon release rate in the cloud system.
- **Degree of Imbalance:** This describes the variation between VMs.
- **Makespan:** It is used to represent the total time or completion time taken to allocate resources to the users.

5.3 Load-Balancing Policies

As discussed above in classification, several techniques are divided into various categories, such as static and dynamic for the state of the system. Dynamic algorithms use state of the system and some policies to execute the tasks (Alakeel 2010; Daraghmi and Yuan 2015; Eager et al. 1986; Kanakala and Reddy 2015; Mukhopadhyay et al. 2010), as follows:

- **Selection Policy:** This policy identifies the tasks that should be transferred from one node to another. It selects the tasks based on the amount of overhead required for migration, the number of non-local system calls, and the time required to execute the task.
- **Location Policy:** This policy determines the computing nodes that are underloaded or free and transfers tasks to them for processing. It selects the destination node after determining the availability of required services for task migration based on available approaches: Probing, Negotiation, and Random. In the random approach, the location policy selects the destination randomly and transfers the tasks. In the probing approach, a node probes other system nodes to select the destination. In the negotiation approach, nodes negotiate with each other for load balancing.
- **Transfer Policy:** This policy discovers the circumstances in which tasks are required to transfer from a local node to another local/remote node. It consists of two approaches—all current tasks and last received task—to identify the tasks to be transferred. In the last received approach, all incoming tasks enter the transfer policy and the task that arrives last will be transferred. Under all current task approaches, transfer policy based on a rule decides that a task needs to transfer (task migration) or processes it locally (task reschedule) and it depends on each node's workload.
- **Information Policy:** This is another policy of dynamic load balancing that keeps all resource information in the system, which is further used by other policies to make their decisions. It decides the time for information collection. Various methods for information collection from the nodes are Agent, Broadcasting, and Centralized polling. In the broadcasting method, all of the nodes broadcast their information, which is accessible by other nodes. The Agent method is currently used by nodes to collect information. Various information policies are the Demand-driven policy, Periodic policy and State Change–driven policy.

All of these dynamic load-balancing policies have some relation where the tasks entering into a system are initially processed by transfer policy. After processing, a policy decides whether the tasks should be transferred to a remote node or not. For the tasks that require transferring, location policy will identify the destination node that is idle or underloaded. If a remote node is not available for execution, the task will be put into a queue for processing locally. Both transfer policy and location policy collect required information from the information policy to make the decision.

6 REVIEW OF EXISTING LOAD-BALANCING TECHNIQUES

We surveyed the literature on existing load-balancing techniques and thoroughly reviewed the selected papers. Load-balancing techniques are categorized mainly into static and dynamic based on the state of the system.

Here, we discuss existing load-balancing techniques:

6.1 Static Load-Balancing Techniques

Static load-balancing techniques do not need knowledge of the current state of a system; they keep only knowledge of system resources such as execution time, memory, storage capacity, and processing power of nodes in advance. Static load balancing does not allow allocation of resources at execution time. These techniques are easy to implement and execute but are useful for smaller system or networks with a smaller number of resources. Because they do not consider the current state of the system, these techniques are not useful for computing systems performing distributed computing. They also do not allow detection of the connected server machine at execution time, which leads to uneven distribution of resources. Some static load-balancing techniques are Round Robin, Min-Min, Max-Min, Throttled, and First-In-First-Out. Some of the static techniques are discussed below.

The dynamic-exchange algorithm for load balancing allows distribution of a finite load on available machines (Houle et al. 2002). Houle et al. (2002) consider static load balancing with fixed load and synchronous communication on processors. It distributed the tasks in unit-sized jobs (called token) in a single-port single-token model using local computation only. It did not allow changing the load before the load was redistributed. It was not useful for dynamically changing loads and processors executing in a distributed environment.

A static load-balancing technique keeps all task-related information available prior to minimizing the waiting time of tasks (Kokilavani et al. 2011). It takes the list of all incoming tasks and determines their execution time. A task with minimum execution time will be executed first and the task with maximum execution time will be processed last. The shortest jobs will be executed first. However, the system may face starvation for some of the tasks, as they require more processing time because they are not in the queue for execution. Static task-based load balancing allows tasks to execute in round-robin manner (Pasha et al. 2014). Each task executes for a time slice and is put in the queue again to execute another process. A load-balancing technique is followed until all processes complete their tasks. This technique is useful for web servers where all requests arriving are of the same type. However, it is not good for the cloud environment, where a process arrives with a different configuration.

6.2 Dynamic Load-Balancing Techniques

As per the discussion in Section 6.1, static techniques do not need knowledge of the current state of the system because these techniques are unsuitable for distributed computing systems that change state dynamically and need the current state of resources at a time interval. Thus, we require dynamic load-balancing techniques that are suitable for the cloud environment. In this section, we discuss various dynamic load-balancing techniques that depend on load balancer criteria. Based on our observation, we have categorized them as follows:

- General Load Balancing
- Natural Phenomena-Based Load Balancing
- Hybrid Load Balancing

- Agent-Based Load Balancing
- Task-Based Load Balancing
- Cluster-Based Load Balancing

6.2.1 General Load-Balancing Techniques. Here, we discuss various general techniques, such as VM migration and load estimation–based algorithms depending on various load-balancing parameters.

Alakeel (2010) has produced a guide to the issues that need to be considered while developing or studying load-balancing algorithms . The author discusses different components of dynamic load balancing, such as information, transfer, and location strategies. The author covers the following issues: load evaluation, load levels comparison, a performance indicator, stability, and amount of information exchange among the nodes. Also, the author discusses the limitation of a cloud load-balancing algorithm, concluding that the current load balancing policies do not keep information about the previous state of the VM while assigning the job to VM. Load-balancing algorithms require running each time a new request comes to a data center for allocation. The author also implemented a round-robin algorithm in Java language using the CloudSim toolkit.

An algorithm to achieve two main goals—avoidance of overloading in the cloud environment and green computing by optimizing used servers—has been proposed by Xiao et al. (2013). The authors introduce the concept of skewness to assess use of servers and improved overall use of resources in the multidimensional constraint. They also propose a load prediction algorithm that helps in identification of future resource consumption by the application. When overall resource use is below the green-computing threshold, it automatically evokes the green-computing algorithm to shut down some of the underutilized servers. The authors have evaluated the algorithm using trace-driven simulation based on various data traces collected from different sources. Their results state that the algorithm has improved overall resource use and minimized power consumption but it can lead to overloading for a few of the resources.

A deadline-constrained scheduling algorithm for aperiodic tasks in an inter-cloud environment has been proposed by Pop et al. (2015). The authors have suggested a method that identifies the number of data resources required to process a set of tasks that are aperiodic in nature. It considers deadline or execution time and data transfer cost as the main constraint for aperiodic task scheduling. From the big-data perspective, the authors have challenged traditional scheduling algorithms for supercomputing and, based on a mathematical model, proved that a set of tasks that are from different sources can be considered as a single one. The authors have also proven that task migration in regional centers is the best solution when a required number of resources are more than a data center's capacity and a higher number of resources are required for a problem in a heterogeneous environment. The results will be helpful in optimizing resource use in data centers.

Wang et al. (2015) propose a framework design to balance the workload on a cloud storage system named as Swift. The authors state that the designed framework was able to identify the nodes that were overloaded and underloaded in the cluster. They had designed a separate algorithm for load balancing and resource allocation to distribute the tasks on VMs that also helps in managing the resources properly. The framework resides on the user side and did not require any changes in user OS and storage systems. It works in the scope of regulation and for a smaller number of nodes only. However, it did not work if the number of nodes was high with several scopes and for different storage systems.

A load-balancing technique is required to estimate the finish time for tasks that considers current processing power and job size on a VM. Chien et al. (2016) present two factors for finish-time estimation: (1) a selected VM should complete the work aforetime and (2) a load-balancing algorithm has to estimate the finish time for the queued job and for the jobs that are coming next into the queue. It will select the VM for job distribution that responds earlier and allocate the task to execute. The results stated that it improves response time and processing time but it leads to increased power consumption and carbon emission.

Bala and Chana (2016) state that to manage load on a cloud proactively during execution, it should be predicted on nodes using a machine-learning approach. They propose a load-prediction model to identify overloaded and underloaded machines to implement proactive load balancing using multiple resource parameters. The authors implemented the model with Random Forest, a machine-learning approach in the Cloudsim tool environment, by considering different resource use parameters to enhance the load balancing in VMs. The proposed model was tested in a simulated environment only; it could be tested in a real-time environment in the future. Also, the model could be enhanced by incorporating fault-tolerance techniques into the proposed prediction model.

Neglia et al. (2016) propose a mean field technique based on load balancing among micro-data centers powered by renewable energy. The technique gives priority to data centers for task execution that are powered by renewable sources. Mean field technique in the proposed model helps to infer various performance measures.

Load-balancing techniques serve to minimize the response time for user requests and assign them uniformly to available resources. Ghoneem and Kulkarni (2017) have modified an active VM load-balancing technique available on a cloud analyst simulator. The existing VM load-balancing technique allocates requests unequally during peak time, which leads to higher response time for users. They used a reservation table along with an allocation table to maintain information regarding VM reservations for the requests. The data center takes into account this information provided by the load balancer and allocates a VM to the upcoming user request. Results state that the proposed technique provides minimum response time to user requests even during heavy traffic.

Chen et al. (2017) present a novel load-balancing approach over a static load-balancing approach mainly considering server processing power and computer loading to minimize the load over servers. They describe the various load-balancing metrics used in then existing algorithms and compare with the proposed algorithm. However, they do not consider the migration time and response time for the tasks, which could be improved with the existing parameters to improve the load balancing of the algorithm. We have thoroughly analyzed and summarized general load-balancing techniques in Table 3.

6.2.2 Natural Phenomena—Inspired Load-Balancing Techniques. Here, some natural phenomena–inspired load-balancing techniques—such as Genetic Algorithm, Honey Bee Inspired, ACO, and PSO—are summarized in Table 4.

A honey bee behavior—inspired load-balancing technique presented by Babu and Krishna (2013) facilitates transfer of a task for execution from overloaded machines to underloaded machines. It also balances priority for a task to be executed on a machine that leads to minimal waiting time for tasks. It works on the concept of honey bees (i.e., tasks) which make a call to other honey bees for the food source (i.e., underloaded machines). It makes priority for tasks the main QoS parameter. The proposed algorithm works for independent tasks that are executing on VMs, but it does not work for dependent tasks. The algorithm can be modified to work for dependent tasks. Other parameters that can be included are execution time and response time.

De Falco et al. (2015a) propose an extremal optimization (EO)–based technique for load balancing of parallel executing user requests in a dynamic environment. A set population-based parallel EO method considers factors such as selection of fitness function and target nodes, which give solutions for dynamic methods. During simulation, the results were compared with the results of sequential EO-based algorithms (De Falco et al. 2014, 2015b). The variant PEO-GS-D based on a

Table 3. Summary of General Load-Balancing Techniques

References	System State	Technique	Concept	Pros	Cons
(Wang et al. 2015)	Dynamic	Workload balancing and resource management framework for Swift Storage	Live VM Migration	Low task execution	Homogeneous VM, Independent task
Chien et al. (2016)	Dynamic	Using estimation method of job finish time	Load balancing VMs using end of service time	Reduced response time, reduced processing time	Actual instant processing power calculation is difficult, more power consumption
Bala and Chana (2016)	Dynamic	Predictive load-balancing approach using machine learning	Identifying overloaded and underloaded machine using machine learning	High resource use, low migration overhead, reduced number of migrations	Not tested on a real cloud
Ghoneem and Kulkarni (2017)	Dynamic	Modified active VM load-balancing technique	Use of reservation table for uniform allocation of requests	Minimum response time for tasks, load balancing among VMs, improves elasticity	Allocates tasks uniformly on single data center
Chen et al. (2017)	Dynamic	Novel load-balancing approach for minimizing load on servers	Dynamic annexed method over static load balancing	Improved resource use, makespan, and QoS	Higher response time and few load-balancing parameters considered

Table 4. Natural Phenomena–Inspired Load-Balancing Techniques

References	System State	Technique	Concept	Pros	Cons
Babu and Krishna (2013)	Dynamic	Honey bee behavior–inspired load balancing	Use of foraging behavior of bees	Low makespan and response time	Does not work for dependent tasks
De Falco et al. (2015a)	Dynamic	Extremal optimization–based load balancing	Parallel task execution in dynamic environment	Lower execution time, llower number of task transfers, higher resource use	Does not support graph optimization and multi-objective optimization
Babu and Samuel (2016)	Dynamic	Enhanced bee colony–based load balancing	Use of honey bee technique to minimize resource consumption and response time	Low response time, high resource use, lower number of task migrations	Low scalability, complexity
(Devi and Uthariaraj 2016)	Dynamic	Weighted round-robin technique	Decreasing response time of tasks while considering execution time	Low response time	Homogeneous environment execution

solution of parallel branches gives the best result with an average case. This algorithm does not work for application graph optimization and multi-objective optimization.

Load-balancing techniques are required for improving resource use, minimizing response time and completion time for tasks on the cloud. Babu and Samuel (2016) propose a technique considering response time, QoS, and number of migrations as load-balancing parameters. They consider the tasks to be honey bees and underutilized VMs to be the food source. When a VM was overloaded, then some of the tasks (tasks with low priority) were migrated from one virtual machine to another. The algorithm could be enhanced with other nature-inspired algorithms, such as PSO and ACO.

Devi and Uthariaraj (2016) present an improved weighted round-robin algorithm for improving the response time of task execution by considering the task size, capacity of the VM, and interdisciplinary nature of multiple tasks. The algorithm considers the current load of all VMs and completion time of executing tasks and identifies the machine with the least completion time. It runs a load balancer after completion of each task to distribute the load among the nodes. The proposed algorithm does not execute in a heterogeneous environment, which could be improved by adding machines from different environments to achieve consistent results. We analyze natural phenomena–inspired load-balancing techniques in Table 4.

6.2.3 *Hybrid Load-Balancing Techniques.*

Hybrid load-balancing techniques are developed to overcome drawbacks of static and dynamic techniques by preserving their features and advantages. Hybrid techniques minimize response time and provide efficient resource use. We summarize hybrid load-balancing techniques in Table 5.

Wang et al. (2013) present a resource scheduling technique for the hybrid cloud. They consider various parameters, such as execution time, finish time, and utilization rate of public and private clouds. Based on these parameters, they propose Adaptive Scheduling with OoS Satisfaction (AsQ) to allocate resources optimally on a private cloud. They use various runtime estimation and scheduling algorithms to identify optimal resource allocation in a private cloud. Using this process, they have reduced the tasks to be transferred to the public cloud for execution. For tasks that need to be transferred, a minimum cost strategy is used to minimize the rate of tasks transferred to the public cloud. They have performed many experiments and the results show that execution time and finish time of tasks are minimized as compared with many existing techniques. Their techniques have better QoS. However, the technique has not considered other important parameters, such as energy efficiency and operational cost, which are needed to revise the model for transferring tasks to the public cloud.

Arab and Sharifi (2014) have developed a communication model for load balancing and resource discovery units. Each node is equipped with load-balancing and resource-discovery units separately. Load-balancing units exchange messages with the same unit in other nodes and get enough information about available resources. This information helps resource-discovery units to make decisions more accurately. According to the proposed model, the load balancer is responsible for extracting the processor's status using various techniques described by Dodonov and De Mello (2010). It provides scalability and lower response time but it leads to increased resource discovery time.

A hybrid load-balancing scheme consisting of on-demand scheduling, Querying and Migrating Task (QMT) and Staged Task Migration (STM) helps to manage the load on nodes more effectively (Liu et al. 2015). Whenever a node detects a heavy load, QMT will balance the load by transferring the last incoming node to another node with a low load. Both QMT and SMT work well for dependent and independent tasks, but they suffer from high transmission and scheduling time that can be minimized in future.

Table 5. Summary of Hybrid Load-Balancing Techniques

References	System State	Technique	Concept	Pros	Cons
Wang et al. (2013)	Dynamic	Adaptive scheduling technique for parallel tasks	Minimum cost strategy used for task transfer to improve resource use	Higher resource use, minimized task migration, better QoS	Important parameters such as operational cost, energy efficiency are not considered.
Arab and Sharifi (2014)	Dynamic	Resource discovery and load-balancing combination	Increase scalability by communication between the resource and load balancing.	Highly scalable, No other node information required	High resource discovery time, propagation of policies
Cho et al. (2015)	Dynamic	ACO combined with PSO	Combined ACO with PSO to improve resource use	Increased resource use, low computation time	High computation cost, homogeneous server support
Liu et al. (2015)	Dynamic	Hybrid load balancing and task scheduling	Balancing load on slave nodes using hybrid load balancing based on master node	Independent and dependent task scheduling, lower response time	High transmission and scheduling time
Naha and Othman (2016)	Dynamic	Combination of broker and load-balancing techniques	Combining load-balancing and broker technique to reduce response time	Low processing time, low response time	Low performance, high execution time
(Chen et al. 2017)	Dynamic	Cloud load-balancing (CLB) technique	Architecture to overcome server response failure and load balancer to monitor priority, computing power	Highly scalable	High response time

Cho et al. (2015) present a hybrid load-balancing technique by incorporating the features of PSO with ACO. Every time a request arrives, it checks the load on each server and identifies the maximum available memory. The proposed technique predicts the upcoming load based on the history of loads on server, which leads to higher computation cost in the case of memory and time due to regular analysis of available memory on servers. Based on availability of memory and size of task, it accepts or rejects the incoming request before scheduling. As a result, accepted requests take turn executing, which keeps all servers busy. It leads to improved resource use and computation time of tasks. This technique conforms the dynamic environment for execution by predicting the workload for coming requests based on historical information about tasks. A pre-reject step rejects the requests that do not fulfill prescheduling criteria, which minimizes computation time. The authors state that the proposed technique is much faster than the traditional ACO technique but that it works only on homogeneous servers and does not consider the cost of client and service providers.

Naha and Othman (2016) have implemented a hybrid algorithm to optimize system performance by combining round-robin and throttled load-balancing techniques with performance-optimized service broker and service proximity broker algorithms. They have proposed three service-broker algorithms and one load-balancing algorithm. The authors have described them as Load

Table 6. Summary of Agent Based Load Balancing Techniques

References	System State	Technique	Concept	Pros	Cons
Chen et al. (2013)	Dynamic	Novel emergent task allocation in the cloud	Emergent task allocation in the cloud using fair competition and dynamic adjustment principle	Efficient resource allocation	Higher processing time, higher transmission time
Tasquier (2015)	Dynamic	Multiagent-based load balancing	Using multiple agents for resource provisioning and monitoring in multi-cloud environment	Provides extreme elasticity facility, uses multi-cloud resources	Not implemented, QoS is not considered
Garcia and Nafarrate (2015)	Dynamic	Agent-based load balancing	VM migration using agent	Heterogeneous VM and server	high migration overhead, low scalability
Keshvadi and Faghih (2016)	Dynamic	Multiagent-based load-balancing architecture	Maximizing resource use with multiple agents	Low response time, improved makespan, improved resource use	DcM agents require parent message to destroy themselves, no timer for self-destroying

Aware (LA) and Cost Aware (CA) algorithms for higher resource use. The LA algorithm provides lower processing time but generates a higher cost while the CA algorithm lowers the cost. The service-broker algorithm selects the server according to user requirements, which may lead to an increase in either processing time or cost. When CA user requests com on the server, they keep the high-speed (high-cost) servers idle, which lowers resource use and increases computation time for user requests. A service proximity broker algorithm selects the data center nearest to the client's region. The throttled load balancer maintains a table for all available VMs. The authors have combined all of the mentioned service-broker and load-balancing algorithms in the pair to serve user requests; the results state that processing time and response time are reduced. However, it takes more execution time for requests and the system performance also needs improvement.

Development of an effective Cloud Load Balancing (CLB) architecture is required to overcome response failure from the server in the case of more user requests. Chen et al. (2017) have developed architecture that considers computer loading and server processing to minimize the problem of server issues to handle more computation requests. They also present a load-balancing technique for the physical and virtual web server to keep information about server loading, priority, and computing power. The architecture provides highly scalable performance, but it leads to increased response time. We have thoroughly summarized hybrid load-balancing techniques in Table 5.

6.2.4 *Agent-Based Load-Balancing Techniques.*

Here, we survey various agent-based load-balancing techniques. The agent works for cloud resource discovery, negotiation, composition, and management. The agent works automatically and continuously to satisfy the design objective. Multiple agents can work together to satisfy user QoS requirements and resource use. In Table 6, we summarize these techniques.

Chen et al. (2013) have developed an emergent task allocation technique for the cloud. To achieve load balancing, the authors employ the fair competitive principle and dynamic adjustment principle to accommodate arriving tasks. The technique uses a roulette wheel mechanism in which the bidder participates in the bidding process to get the task to allocate to a resource and uses a buffer pool mechanism to strengthen the performance of the approach. Experimental results show that the techniques allocate resources efficiently but increase the processing and transmission times.

Application aware, multiagent-based load-balancing architecture facilitates provisioning of resources automatically. Tasquier (2015) states that the developed architecture uses three different agents—executor agent, provisioner agent, and monitor agent—which are responsible for representation of running applications, scaling in and out of resources, and monitoring of overload/underload conditions of resources, respectively. The proposed algorithm also provides the facility to review the current state of cloud and resource elasticity. However, the author has not implemented the algorithm in the cloud environment to test validity and does not consider QoS.

Garcia and Nafarrate (2015) present a collaborative agent–based load-balancing technique that handles the load across heterogeneous servers using a live VM migration concept. They also propose an agent-based load-balancing mechanism that consists of a program that identifies the VMs that need to be migrated along with the destination, strategies for VM migration, acceptance policies for VMs, and a load-balancing program to select the initial host for VMs. The authors state that the algorithm performs better in load balancing as comparing with centralized techniques that improved response time for the tasks as well as resource use, but it increases the migration overhead.

Multiagent-based load-balancing architecture helps in maximizing resource use (Keshvadi and Faghih 2016). It executes both sender-originated and receiver-originated techniques to reduce a task's waiting time and to ensure SLA. The model consists of the following agents: the Virtual Machine Monitor Agent (VMM Agent), Datacenter Monitor Agent (DcM Agent) and Negotiator Ant Agent (NA Agent). The VMM agent supports all VMs in the system and keeps information on memory, CPU, and bandwidth use by VMs to monitor load. The DcM agent performs Information Policy using information available from the VMM agent and categorizes VMs depending on different characteristics. It also initiates NA agents, which move to other data centers to know the status of VMs available there. The simulation results state that it is more efficient and improves response time and makespan. We analyze agent-based load-balancing techniques in Table 6.

6.2.5 *Task-Based Load-Balancing Algorithms.*

Here, we discuss the literature on task-based load-balancing techniques. Most of the existing load-balancing schemes prefer to migrate VMs by transferring an overloaded VM from a physical machine to another physical machine. It enables flexible resource allocation but requires more time and cost, as we need to transfer the whole VM machine rather than transfer the tasks owing to which machines are getting overloaded. Table 7 summarizes existing task-based load-balancing techniques.

Ramezani et al. (2014) have developed a Task-Based System Load-Balancing (TBSLB) approach. The approach is based on the PSO technique to reduce the migration cost of VMs. They have also developed an optimization model for transferring tasks. They evaluate the proposed model by extending CloudSim embedded with the Jswarm package (Calheiros et al. 2011). The proposed mechanism has benefits over then-existing load-balancing techniques in that it minimizes the downtime for VMs, memory use, and cost. They have worked on the parameters transfer time, memory downtime, and usage cost. Later, it can be improved for multi-objective PSO and for other load-balancing parameters.

Wu et al. (2016) have proposed a genetic and ant colony–based task-scheduling algorithm. They combine the features of both genetic and ACO techniques. A genetic algorithm searches for the available resources at an earlier stage and then the ant colony algorithm selects the optimal resource for the tasks. Individually, the genetic algorithm has more response time, lower efficiency, and redundancy. The ant colony algorithm also does not work well at resource search stage owing to a lack of pheromone. By combining features of both algorithms, the hybrid approach improves load balancing among VMs and increases the efficiency.

Table 7. Summary of Task-Based Load-Balancing Techniques

References	System State	Technique	Concept	Pros	Cons
Ramezani et al. (2014)	Dynamic	Task-based system load-balancing method using particle swarm optimization	Live VM migration	Low task execution	Homogeneous VM, independent task
Wu et al. (2016)	Dynamic	Genetic–ant colony hybrid algorithm for task scheduling	Combine features of genetic and ant colony optimization technique Genetic search for available resources and ant colony selects optimal solution for execution.	Improved load balancing, increased efficiency, minimized execution time	Not tested in actual cloud environment
Shen et al. (2016)	Dynamic	Considering network topology and transmission time in scheduling	Implementing task scheduling on MapReduce to minimize data transmission time and cost	Improved cluster use, minimized job completion time	No bandwidth reservation, model is not evaluated under different network condition
Elmougy et al. (2017)	Dynamic	A hybrid task scheduling algorithm based on shortest job and round robin with dynamic task quantum	Storing short and long tasks in separate ready queues and executing separately to minimize starvation	Starvation and waiting time for tasks minimized, improved response time and turnaround time	Task quantum is less effective
Xin et al. (2017)	Dynamic	Cost efficient multiple schedulers for parallel tasks	Multiple scheduler to execute parallel tasks and weighted machine allocation approach	Improved resource use, minimized task weighting and execution time	Parameter evaluation is not optimal

Shen et al. (2016) present a network-aware task placement technique for MapReduce to reduce job completion time, overall transmission time, and cost of data. They state that a task mainly faces the following problems: first, resource availability changes dynamically owing to release and access over time; second, data-fetching time for reduced tasks depends on size and location of tasks; and third, the load on the path also has a relevant effect on data access latency. To minimize data access latency, load over the path should be considered during scheduling decisions for the tasks. Results show that it has minimized completion time for tasks and increased resource use.

A multiple-scheduler architecture facilitates minimizing the problem of executing large-scale parallel tasks and their execution cost. Xin et al. (2017) propose a scheduling method to minimize resource competition between tasks and high device load, which is based on weighted random scheduling. The tasks will be assigned weights considering parameters such as execution time, cost, and communication delay. A machine with lower cost will get higher cost and have more chances to be assigned for task execution. The authors have experimented with MATHLAB2012b using WorkFlow generator to generate the required dataset. They have experimented on datasets that include the larger set of tasks in an interval with execution time, cost, and transmission delay. They have also considered task structure, task arrival time, device dependency, and specific device set test. The results show that multiple schedulers have improved parameters such as task competition for devices and execution cost. However, it has not evaluated optimal value for the parameters, which can be further improved.

The Selecting VM with Least Load (SVLL) load-balancing model for task distribution has increased the performance of cloud systems (Aladwani 2017). The proposed model calculates the load of each VM and assigns the tasks for execution based on the load of the VM rather than the number of tasks assigned to the VM. Aladwani (2017) has implemented the SVLL technique with other task-scheduling techniques such as first-come-first-serve and shortest job first, which results in improved total waiting time and total finish time of tasks. The algorithm is implemented with the basic scheduling algorithms, which may give a better result as compared to the basic algorithms alone.

Elmougy et al. (2017) have developed a task load-balancing technique combining features of the shortest job first and round-robin scheduling techniques. It stores short and long tasks in two separate ready queues and uses dynamic task quantum to balance waiting time among the tasks. The authors have considered the issue of throughput and starvation together. They have evaluated the algorithm on the CloudSim simulator; the results show that turnaround time, waiting time, and response time are minimized. It has also minimized the long task starvation. However, the task quantum is not very effective in balancing among tasks that can be improved in the future to improve waiting time and the starvation problem. We analyze task-based load-balancing techniques in Table 7.

6.2.6 Cluster-Based Load-Balancing Techniques. In the heterogeneous cloud environment, various resources are placed at different data centers and are divided into different clusters based on parameters such as the performance of servers and storage capacity. In this section, we discuss some cluster-based techniques and summarize them in Table 8.

A prediction-based adaptive load-balancing technique helps in improving data center resource use. Mao et al. (2013) state that the proposed technique predicts the load in clusters and accordingly callback resources in a cluster if the workload is below a minimum threshold and add new VMs when the workload is above the threshold. The authors have implemented the algorithm on the Cloudsim simulator; the results show that it has improved resource use and has minimized response time for the tasks.

Daraghmi and Yuan (2015) propose a load-balancing mechanism for developed controllers to overcome the issue limitations of a centralized controller in mega data centers. It divides the whole network into many regions and allocates a controller in each region to reroute the flows through analyzing imbalanced load issues in developed controllers for traffic control and network management. In the proposed technique, when the imbalanced load occurs, it migrates a portion of the load to the controller to dynamically manage the load. The authors have developed multiple solutions for the LBDC technique, including greedy approaches and one distributed greedy approach to manage the traffic and compared the results with existing developed controllers.

Zhao et al. (2016) address a heuristic technique for load balancing based on Bayes and Clustering (LB-BC) to overcome the complexities of existing load-balancing approaches. The approach is based on Bayes' theorem (Agostini 1995; Pawlak 2001) and has achieved long-term load balancing. It calculates the posteriori probability of physical hosts and combines it with the clustering idea to pick an optimal host. It considers these parameters: number of requested tasks, standard deviation, and load-balancing effect. Later, the approach was compared with dynamic load balancing, which leads to minimum standard deviation with increased time. The proposed approach works in a local area only but can be enhanced for wide area networks and real-time environments.

Kang and Choo (2016) present a cluster-based job-dispatching technique to improve inter-cloud communication for load balancing in dynamic and real-time multimedia streaming. The proposed technique has a two-step process. First, it creates the cluster for monitoring activities, managing

Table 8. Summary of Cluster-Based Load-Balancing Techniques

References	System State	Technique	Concept	Pros	Cons
Daraghmi and Yuan (2015)	Dynamic	A small world-based overlay network for dynamic load balancing improvement	Greedy approach-based developed controller in decentralized mega data centers	Improved resource use, increased data center performance	High power consumption
Kang and Choo (2016)	Dynamic	Cluster-based job-dispatching technique	Improving large-scale inter-cloud communication for load balancing in dynamic and real-time multimedia streaming	Better response time	Packet loss owing to congestion
Zhao et al. (2016)	Dynamic	Load balancing based on Bayes and clustering (LB-BC)	Posteriori probability of physical hosts and combines it with the clustering concept	Minimum standard deviation and response time	Good for LAN only, not applicable for real-time environment
Han and Chronopoulos (2017)	Dynamic	Distributed model for self-scheduling techniques to improve load balancing	Class of distributed self-scheduling schemes to improve load balancing and scalability	Improved scalability, improved overall performance, reduced communication overhead	Parallel execution not supported

platform complexities, and meeting demands and satisfactory QoS for the hosts on the basis of a hello packet broadcast to all connected neighbor servers periodically. Second, it makes a decision to transfer the job requests if the waiting time for the job is more than 5 tasks in the queue. Performance of this technique is compared with ant colony, WCAP, and HFA along with the higher rate of job dispatching and it gives better response time. Further, this technique can be improved for the real-time environment where intermediate nodes get congested owing to resource limitations and to reduce data loss due to congestion using communication jobs rather than computation jobs.

Zegrari et al. (2016) present a cluster-based task load-balancing technique to overcome the problem of load distribution on the nodes. It combines the concept of genetic and KUHN algorithms and developed task allocation strategy by grouping tasks into clusters and distributing them in cooperating nodes. The proposed algorithm provides better response time and task distribution among the data center nodes.

Han and Chronopoulos (2017) have developed a hierarchical distributed model for self-scheduling schemes to improve load balancing and scalability of the cloud system. The model is able to execute both in homogeneous and heterogeneous environments. The authors have applied the schemes in a large-scale cluster using four different computation applications; results show improved scalability, improved overall performance, and reduced communication overhead. In future, the algorithm can be tested for large-scale clusters and loops with dependencies. We analyze cluster-based load-balancing techniques in Table 8.

6.2.7 Load Balancing and Resource Scheduling Tools. In this section, we discuss some of the available cloud simulators (Calheiros et al. 2011; Garg and Buyya 2011; Gupta et al. 2011; Jararweh et al. 2013; Nunez et al. 2011; Ostermann et al. 2010; Wickremasinghe et al. 2010). A cloud simulator is a framework that provides a virtual environment to test performance, deployment models, resource allocation, and load balancing of resources when development of real cloud architecture for evaluation of these factors is not possible owing to the cost factor. Simulators play a major role in testing and validating load balancing, scheduling algorithms before deploying on real hardware. All simulators have some unique features and are used to evaluate different parameter performance. Table 9 shows the feature, availability, programming language used, strength, and weakness for available simulators.

7 DISCUSSION

We provide a comparative analysis of studied articles based on various metrics used by these techniques in the cloud computing environment shown in Table 10 and Figure 5. Table 10 shows how various research papers have considered different QoS metrics. After reviewing the various load-balancing techniques comprehensively, it can be stated that different techniques have considered different metrics for evaluation. Some of the papers have considered a single objective while some have considered multiple objectives for metrics. Figure 5 shows the percentage of load-balancing metrics considered by different articles. Figure 6 shows the evaluation tools used in different research papers.

We also have analyzed the various research papers based on simulations and results available. Table 10 and Figure 5 show that most users have concentrated on response time to the tasks coming up for execution, followed by resource use, makespan, and migration time. Researchers are focusing on minimization of response time to maintain the SLA and increasing the response time. Devi and Uthariaraj (2016), Babu and Krishna (2013), and De Falco et al. (2015a) have considered response time to be the main parameter for evaluation of algorithms.

8 RESEARCH TRENDS

In cloud load balancing, there are still many issues and challenges that need to be discussed and resolved in future. From the literature review, we have found some of the future directions where the cloud needs improvement. To maintain cloud performance some future directions need to be considered: Quality of Service (QoS), the Service Level Agreement, Resource Provisioning, and Load Balancing, among other issues. A number of resources are required of cloud service providers to maintain QoS and the SLA. SLAs are designed and deployed based on QoS rules and if there is any violation of the SLA then a service provider must pay a penalty. Automatic resource provisioning minimizes user and service provider interaction. To maintain the SLA and QoS, load balancing is required for proper use of provisioned resources. A load balancer helps to keep the amount of resources minimum with high throughput. Many load-balancing techniques have been developed that consider different metrics, such as performance, response time, execution time, task migration time, and resource use. No technique has considered all load-balancing parameters, which improves the overall performance of the data centers.

There are important open issues and challenges in cloud load balancing, as follows:

- Proper allocation of resources to a workload is required to improve cloud performance.
- Both functional requirements (SLAs) and non-functional requirements (QoS) should be maintained.
- Managing an instant occurrence of a large number of user requests along with managing the SLA for currently executing tasks.
- Maintaining the cost of application in the case of execution with different service providers and heterogeneous environments.

Table 9. Summary of Resource Load-Balancing and Scheduling Tools

Simulator	Description	Underlying Platform	Programming Language	Availability	Strength	Weakness
CloudSim	It is the most commonly used simulator. It provides a simulation environment to model a cloud or network of clouds. It provides the facility to implement load-balancing and scheduling algorithms by extending Java classes.	SimJava	Java	Open Source	Most commonly used simulator, supports large-scale problems	Packet level communication is not supported
Cloud Analyst	It extends the functionality of CloudSim and provides full GUI-based environment. It allows testing large applications with varying parameters.	CloudSim	Java	Open Source	Focuses on geographical factors, supports service-broker and load-balancing policies implementation	No full network model, fulfills specific purpose only
Network CloudSim	It is an extension of CloudSim. It supports cloud network modeling and resource allocation.	CloudSim	Java	Open Source	Supports full application model, message passing for communication	Packet-level networking not supported, less focus on network cost and power model
Teach Cloud	It is GUI-based extended model of CloudSim. Developed for academic purposes.	CloudSim	Java	Open Source	Simple and easy to use GUI-based tool for educational purposes	Supports basic features only
CDOSim	Cloud Deployment Simulator (CDOSim) is an extension of Cloudsim; helps in deploying optimized cloud according to user's requirement. It allows one to choose a cloud service provider, deployment policies, and resource configuration.	CloudSim	Java	Academic	Provides cloud deployment functionalities, helps to identify and migrate to another cloud	For large-scale and general-purpose cloud testing it is unproved
GDCSim	GDCSim (Green Data Center Simulator) is designed to evaluate energy-efficient techniques in a data center.GDCSim gives a better result as compared to existing simulators based on conditions such as scheduling algorithms, power distribution schemes, and workload distribution.	BlueSim	Java	Academic	Focuses on power saving and cooling methods, evaluates energy-efficient techniques in data center	Not proved for large-scale and general-purpose testing

(Continued)

Table 9. Continued.

Simulator	Description	Underlying Platform	Programming Language	Availability	Strength	Weakness
iCanCloud	It is a SimCan-based fully GUI simulator. It allows users to configure VMs, data center for different cloud scenarios. It also supports service brokering, storage models, and cost prediction in the data center.	OMNET	C++	Open Source	Fully GUI-based simulator,supports storage modeling, service brokering, live migration	No focus on energy consumption
MDCSim	Multi-tier Data Centre Simulator supports system evaluation and power consumption measurement.	CSIM	Java/ C++	Commercial	General purpose functionalities are supported, supports different configuration evaluations	No full network model, commercial
Green Cloud	It evaluates the scheduling algorithms for energy efficiency in the data center by calculating power consumption of servers, network switches, and other communication links.	CloudSim	Java	Open Source	Evaluates energy-efficient techniques in the data center	Not proved for general-purpose components

Table 10. Load-Balancing Metrics in Surveyed Techniques

No.	References	Response Time	Migration Time	Throughput	Makespan	Resource Use	Scalability	Fault Tolerance	Cost
1	Wang et al. (2013)	✓	✓	-	✓	✓	-	-	✓
2	Mao et al. (2013)	✓	-	✓	-	✓	-	-	-
3	Hsueh et al. (2014)	✓	-	-	-	-	-	-	-
4	Garcia and Nafarrate (2015)	✓	-	✓	-	-	-	-	-
5	Pacini et al. (2015)	✓	✓	✓	-	✓	-	-	-
6	Pop et al. (2015)	✓	-	-	-	✓	-	-	✓
7	Yakhchi et al. (2015)	-	✓	✓	-	✓	-	-	-
8	Domanal and Reddy (2013)	✓	-	-	✓	✓	✓	-	-
9	(Babu and Samuel 2016)	✓	-	-	-	-	-	-	✓
10	Keshvadi and Faghih (2016)	✓	✓	✓	-	✓	-	-	-
11	Wu et al. (2016)	✓	-	-	✓	-	-	-	-
12	Kianpisheh et al. (2016)	-	-	-	✓	✓	-	-	-
13	Ghosh and Banerjee (2016)	✓	-	-	✓	-	-	-	-
14	Bok et al. (2016)	-	-	-	✓	-	-	-	-
15	Singh et al. (2016)	✓	✓	-	✓	✓	✓	-	-
16	Yang et al. (2016)	✓	-	-	✓	-	-	✓	-
17	Salehi et al. (2016)	-	✓	✓	✓	✓	-	-	-
18	Zegrari et al. (2016)	✓	-	-	-	✓	-	-	-
19	Chen et al. (2017)	✓	-	-	-	✓	-	-	-
20	Xin et al. (2017)	✓	-	-	-	✓	-	-	-
21	Elmougy et al. (2017)	✓	-	✓	-	✓	-	-	-
22	Ghoneem and Kulkarni (2017)	-	-	-	✓	-	✓	-	-
23	Benifa and Dejey (2017)	-	-	-	✓	✓	✓	-	✓
24	Rodriguez and Buyya (2017)	✓	-	-	-	✓	-	-	-
25	Kaur and Mehta (2017)	-	-	-	✓	✓	-	-	-
	Total	18	5	7	12	15	4	1	4

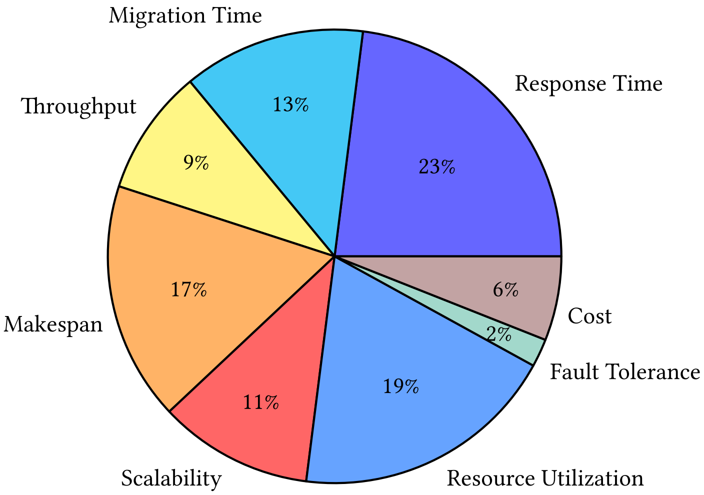

Fig. 5. Percentage of load-balancing metrics in surveyed techniques.

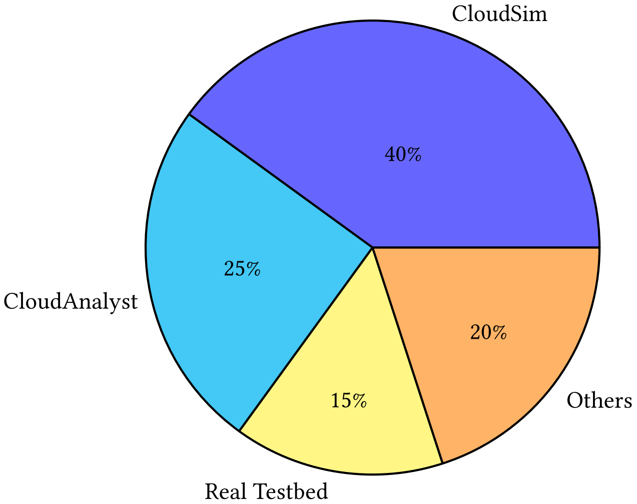

Fig. 6. Evaluation tools/techniques used in survey papers.

- A large number of service providers are establishing data centers. It is a bigger challenge for users to identify the perfect service provider according to their requirements.
- With growing cloud providers, there may be a situation that requires transferring a workload to another cloud provider and may be a challenge owing to different data and service policies.
- Data lock-in can also be an issue when a workload needs to transfer to another cloud provider which requires some policies to resolve such issues.
- Improving the reliability of a cloud system through component-level testing and a checkpoint-based approach.
- Due to the increasing demand for cloud service along with data centers, power consumption is also increasing; minimizing power consumption is a significant issue.

- A huge amount of data is generated daily from different sources, such as banking, social sites, and e-commerce, which requires a high-quality storage system and a stellar technique for easy retrieval and analysis.
- Management of resources and applications in the heterogeneous cloud environment is a very complex task.
- A well-designed resource allocation mechanism is a significant issue for service providers to keep improving resource use.

9 CONCLUSION

Load balancing of tasks on VMs is a prime challenge in cloud computing that has commanded significant attention from researchers. This article represents a state-of-the-art review of issues and challenges of load balancing. According to this study, an extensive survey has been done on numerous load-balancing techniques considering different metrics. Based on our knowledge, we have classified these load-balancing techniques into several categories: General Load-Balancing Techniques, Natural Phenomena–Based Load Balancing, Task-Based Load Balancing, and Agent-Based Load Balancing. From each category, we have discussed advantages, disadvantages, concepts, and challenges with these techniques. Numerous load-balancing algorithms are available that incorporate most load-balancing metrics and provide higher resource use and less response time. However, there is a need to improve the techniques for increasing performance of the system in the future. Along with the improvement in system performance and resource use, load-balancing techniques are also stressing green computing, energy saving, and task load management, which requires developing new algorithms. Therefore, it also becomes necessary to analyze newly developed and recent load-balancing techniques from different categories on simulators based on various load-balancing metrics to check the effectiveness of these algorithms before deployment in the actual cloud environment.

This study will be helpful for researchers to identify research problems working in the load-balancing field and will provide a summary of available load-balancing techniques.

REFERENCES

Shafi Muhammad Abdulhamid, Abd Latiff, Muhammad Shafie, and Mohammed Bakri Bashir. 2014. Scheduling techniques in on-demand grid as a service cloud: A review. *Journal of Theoretical & Applied Information Technology* 63, 1 (2014), 10–20.

G. D. Agostini. 1995. A multidimensional unfolding method based on Bayes' theorem. *Nuclear Institute and Methods in Physics Research, A* 362, 2–3 (1995), 487–498. DOI: https://doi.org/10.1016/0168-9002(95)00274-X

Ishfaq Ahmad and Arif Ghafoor. 1991. Semi-distributed load balancing for massively parallel multicomputer systems. *IEEE Transactions on Software Engineering* 17, 10 (1991), 987–1004. DOI: https://doi.org/10.1109/32.99188

Tahani Aladwani. 2017. Impact of selecting virtual machine with least load on tasks scheduling algorithms in cloud computing. In *Proceedings of 2nd International Conference on Big Data, Cloud and Applications (BDCA'17)*. ACM, New York, NY, Article 13, 7 pages. DOI: https://doi.org/10.1145/3090354.3090367

Ali Mohammad Alakeel. 2010. A guide to dynamic load balancing in distributed computer systems. *International Journal of Computer Science and Information Security* 10, 6 (2010), 153–160.

Mohammad Norouzi Arab and Mohsen Sharifi. 2014. A model for communication between resource discovery and load balancing units in computing environments. *Journal of Supercomputing* 68, 3 (2014), 1538–1555. DOI: https://doi.org/10.1007/s11227-014-1124-y

Dhinesh Babu and Venkata Krishna. 2013. Honey bee behavior inspired load balancing of tasks in cloud computing environments. *Applied Soft Computing Journal* 13, 5 (2013), 2292–2303. DOI: https://doi.org/10.1016/j.asoc.2013.01.025

Remesh Babu and Philip Samuel. 2016. Enhanced bee colony algorithm for efficient load balancing and scheduling in cloud. *Innovation in Bio-Inspired Computing and Application* 4 (2016), 135–142. DOI: https://doi.org/10.1007/978-3-319-28031-8_6

Anju Bala and Inderveer Chana. 2016. Prediction-based proactive load balancing approach through VM migration. *Engineering with Computers* 32, 4 (2016), 1–12. DOI: https://doi.org/10.1007/s00366-016-0434-5

Sourav Banerjee, Mainak Adhikari, Sukhendu Kar, and Utpal Biswas. 2015. Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud. *Arabian Journal for Science and Engineering* 40, 5 (2015), 1409–1425. DOI:https://doi.org/10.1007/s13369-015-1626-9

Bibal Benifa and Dejey. 2017. Performance Improvement of MapReduce for heterogeneous clusters based on efficient locality and replica aware scheduling (ELRAS) strategy. *Wireless Personal Communications* 95, 3 (2017), 2709–2733. DOI:https://doi.org/10.1007/s11277-017-3953-5

Aditya Bhardwaj and Challa RamaKrishna. 2018. Efficient multistage bandwidth allocation technique for virtual machine migration in cloud computing. *Journal of Intelligent & Fuzzy Systems* 36 (2018), 1–14. DOI:https://doi.org/10.3233/JIFS-169819

Kyoungsoo Bok, Jaemin Hwang, Jongtae Lim, Yeonwoo Kim, and Jaesoo Yoo. 2016. An efficient MapReduce scheduling scheme for processing large multimedia data. *Multimedia Tools and Applications* 76, 16 (2016), 17273–17296. DOI:https://doi.org/10.1007/s11042-016-4026-6

Evelyn Brown. 2011. *Final Version of NIST Cloud Computing Definition Published*. Retrieved November 10, 2017 from https://www.bluepiit.com/blog/different-types-of-cloud-computing-service-models/.

Rajkumar Buyya, James Broberg, and Andrzej M. Goscinski. 2010. *Cloud Computing: Principles and Paradigms*. Vol. 87. John Wiley & Sons.

Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya. 2011. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience* 41, 1 (2011), 23–50. DOI:https://doi.org/10.1002/spe.995

Goran Candrlic. 2013. *What Can I Do With It: Cloud Service*. Retrieved February 20, 2018 from https://www.globaldots.com/cloud-computing-types-of-cloud/.

Yeganeh Charband and Nima Jafari Navimipour. 2016. Online knowledge sharing mechanisms: A systematic review of the state of the art literature and recommendations for future research. *Information Systems Frontiers* 18, 6 (2016), 1131–1151. DOI:https://doi.org/10.1007/s10796-016-9628-z

Chao Chen, Xiaomin Zhu, Weidong Bao, Lidong Chen, and Kwang Mong Sim. 2013. An agent-based emergent task allocation algorithm in clouds. In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications & IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC'13)*. IEEE, 1490–1497.

Shang liang Chen, Yun yao Chen, and Suang hong Kuo. 2017. CLB: A novel load balancing architecture and algorithm for cloud services. *Computers and Electrical Engineering* 58 (2017), 154–160. DOI:https://doi.org/10.1016/j.compeleceng.2016.01.029

Nguyen Khac Chien, Nguyen Hong Son, and Ho Dac Loc. 2016. Load balancing algorithm based on estimating finish time of services in cloud computing. In *Proceedings of the 18th IEEE International Conference on Advanced Communication Technology (ICACT'16)*. IEEE, 228–233.

Matin Chiregi and Nima Jafari Navimipour. 2016. A new method for trust and reputation evaluation in the cloud environments using the recommendations of opinion leaders' entities and removing the effect of troll entities. *Computers in Human Behavior* 60 (2016), 280–292. DOI:https://doi.org/10.1016/j.chb.2016.02.029

Keng-Mao Cho, Pang-Wei Tsai, Chun-Wei Tsai, and Chu-Sing Yang. 2015. A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing. *Neural Computing and Applications* 26, 6 (2015), 1297–1309. DOI:https://doi.org/10.1007/s00521-014-1804-9

Biagio Cosenza, Gennaro Cordasco, Rosario De Chiara, and Vittorio Scarano. 2011. Distributed load balancing for parallel agent-based simulations. In *Proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP'11)*. IEEE, 62–69.

Eman Yasser Daraghmi and Shyan-ming Yuan. 2015. A small world based overlay network for improving dynamic load-balancing. *Journal of Systems & Software* 107 (2015), 187–203. DOI:https://doi.org/10.1016/j.jss.2015.06.001

Suman Das, Harish Viswanathan, and Gee Rittenhouse. 2003. Dynamic load balancing through coordinated scheduling in packet data systems. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications, IEEE Societies*, Vol. 1. IEEE, 786–796.

Kousik Dasgupta, Brototi Mandal, Paramartha Dutta, Jyotsna Kumar Mandal, and Santanu Dam. 2013. A genetic algorithm (GA) based load balancing strategy for cloud computing. *Procedia Technology* 10 (2013), 340–347. DOI:https://doi.org/10.1016/j.protcy.2013.12.369

Chitra Devi and Rhymend Uthariaraj. 2016. Load balancing in cloud computing environment using improved weighted round robin algorithm for nonpreemptive dependent tasks. *The Scientific World Journal* 2016 (2016), 1–14. DOI:https://doi.org/10.1155/2016/3896065

Barkat Dhillon. 2015 (accessed February 23, 2018). *Different Types of Cloud Computing Service Models*. Retrieved February 23, 2018 from https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published/.

Evgueni Dodonov and Rodrigo Fernandes De Mello. 2010. A novel approach for distributed application scheduling based on prediction of communication events. *Future Generation Computer Systems* 26, 5 (2010), 740–752. DOI: https://doi.org/10.1016/j.future.2009.05.004

Shridhar Domanal and Ram Mohana Reddy. 2013. Load balancing in cloud computing using modified throttled algorithm. In *Proceedings of the IEEE International Conference on Cloud Computing in Emerging Markets (CCEM'13)*. IEEE, 1–5. DOI: https://doi.org/10.1109/CCEM.2013.6684434

Derek Eager, Edward Lazowska, and John Zahorjan. 1986. Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering* 26, 5 (1986), 662–675. DOI: https://doi.org/10.1016/j.future.2009.05.004

Samir Elmougy, Shahenda Sarhan, and Manar Joundy. 2017. A novel hybrid of shortest job first and round robin with dynamic variable quantum time task scheduling technique. *Journal of Cloud Computing* 6, 1 (2017), 12. DOI: https://doi.org/10.1186/s13677-017-0085-0

Ivanoe De Falco, Eryk Laskowski, Richard Olejnik, Umberto Scafuri, Ernesto Tarantino, and Marek Tudruj. 2014. Extremal optimization with guided state changes in load balancing of distributed programs. In *Proceedings of 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE, 228–231. DOI: https://doi.org/10.1109/PDP.2014.56

Ivanoe De Falco, Eryk Laskowski, Richard Olejnik, Umberto Scafuri, Ernesto Tarantino, and Marek Tudruj. 2015a. Extremal optimization applied to load balancing in execution of distributed programs. *Applied Soft Computing Journal* 30 (2015), 501–513. DOI: https://doi.org/10.1016/j.asoc.2015.01.048

Ivanoe De Falco, Eryk Laskowski, Richard Olejnik, Umberto Scafuri, Ernesto Tarantino, and Marek Tudruj. 2015b. Extremal optimization applied to load balancing in execution of distributed programs. *Applied Soft Computing* 30 (2015), 501–513. DOI: https://doi.org/10.1016/j.asoc.2015.01.048

Octavio Gutierrez Garcia and Adrian Ramirez Nafarrate. 2015. Agent-based load balancing in cloud data centers. *Cluster Computing* 18, 3 (2015), 1041–1062. DOI: https://doi.org/10.1007/s10586-015-0460-x

Saurabh Kumar Garg and Rajkumar Buyya. 2011. Networkcloudsim: Modelling parallel applications in cloud simulations. In *Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing (UCC'11)*. IEEE, 105–113. DOI: https://doi.org/10.1109/UCC.2011.24

Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. 2013. A framework for ranking of cloud computing services. *Future Generation Computer Systems* 29, 4 (2013), 1012–1023. DOI: https://doi.org/10.1016/j.future.2012.06.006

Einollah Jafarnejad Ghomi, Amir Masoud Rahmani, and Nooruldeen Nasih Qader. 2017. Load-balancing algorithms in cloud computing: A survey. *Journal of Network and Computer Applications* 88 (2017), 50–71. DOI: https://doi.org/10.1016/j.jnca.2017.04.007

Mohammad Ghoneem and Lalit Kulkarni. 2017. An adaptive MapReduce scheduler for scalable heterogeneous systems. In *Proceedings of the International Conference on Data Engineering and Communication Technology*. Springer, 603–611. DOI: https://doi.org/10.1007/978-981-10-1678-3_57

Rahul Ghosh, Francesco Longo, Vijay K. Naik, and Kishor S. Trivedi. 2013. Modeling and performance analysis of large scale IaaS clouds. *Future Generation Computer Systems* 29, 5 (2013), 1216–1234. DOI: https://doi.org/10.1016/j.future.2012.06.005

Soumi Ghosh and Chandan Banerjee. 2016. Priority based modified throttled algorithm in cloud computing. In *Proceedings of International Conference on Inventive Computation Technologies (ICICT'16)*, Vol. 3. IEEE, 1–6.

Shipra Goyal and Manoj Kumar Verma. 2016. Load balancing techniques in cloud computing environment–a review. *International Journal of Advanced Research in Computer Science and Software Engineering* 6, 4 (2016), 583–588.

Sandeep Gupta, Rose Robin Gilbert, Ayan Banerjee, Zahra Abbasi, Tridib Mukherjee, and Georgios Varsamopoulos. 2011. Gdcsim: A tool for analyzing green data center design and resource management techniques. In *Proceedings of the International Green Computing Conference and Workshops (IGCC'11)*. IEEE, 1–8.

Hakan Hacigumus, Bala Iyer, and Sharad Mehrotra. 2002. Providing database as a service. In *Proceedings of the 18th IEEE International Conference on Data Engineering*. IEEE, 29–38. DOI: https://doi.org/10.1109/ICDE.2002.994695

Yiming Han and Anthony Theodore Chronopoulos. 2017. Scalable loop self-scheduling schemes for large-scale clusters and cloud systems. *International Journal of Parallel Programming* 45, 3 (2017), 595–611. DOI: https://doi.org/10.1007/s10766-016-0434-5

Michael Houle, Antonois Symvonis, and David Wood. 2002. Dimension-exchange algorithms for load balancing on trees. *Sirocco* (2002), 181–196. DOI: https://doi.org/10.1.1.18.2926

Sue Chen Hsueh, Ming Yen Lin, and Yi Chun Chiu. 2014. A load-balanced MapReduce algorithm for blocking-based entity-resolution with multiple keys. In *Proceedings of the 12th Australasian Symposium on Parallel and Distributed Computing-Volume 152*. Australian Computer Society, 3–9.

Kai Hwang, Jack Dongarra, and Geoffrey C. Fox. 2013. *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. Morgan Kaufmann.

Igor N. Ivanisenko and Tamara A. Radivilova. 2015. Survey of major load balancing algorithms in distributed system. In *Information Technologies in Innovation Business Conference (ITIB'15)*. IEEE, 89–92.

Nima Jafari, Amir Masoud Rahmani, Ahmad Habibizad Navin, and Mehdi Hosseinzadeh. 2015. Expert cloud: A cloud-based framework to share the knowledge and skills of human resources. *Computers in Human Behavior* 46 (2015), 57–74.

Yaser Jararweh, Zakarea Alshara, Moath Jarrah, Mazen Kharbutli, and Mohammad N. Alsaleh. 2013. Teachcloud: A cloud computing educational toolkit. *International Journal of Cloud Computing* 2, 2–3 (2013), 237–257.

Brendan Jennings and Rolf Stadler. 2015. Resource management in clouds: Survey and research challenges. *Journal of Network and Systems Management* 23, 3 (2015), 567–619.

Mala Kalra and Sarbjeet Singh. 2015. A review of metaheuristic scheduling techniques in cloud computing. *Egyptian Informatics Journal* 16, 3 (2015), 275–295.

Ravi Teja Kanakala and Vuyyuru Krishna Reddy. 2015. Performance analysis of load balancing techniques in cloud computing environment. In *Proceedings of the IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT'15)*. IEEE, 1–6.

Byungseok Kang and Hyunseung Choo. 2016. A cluster-based decentralized job dispatching for the large-scale cloud. *EURASIP Journal on Wireless Communications and Networking* 2016, 25 (2016), 1–8.

Nidhi Jain Kansal and Inderveer Chana. 2012. Cloud load balancing techniques: A step towards green computing. *International Journal of Computer Science Issues* 9, 1 (2012), 238–246.

Mayanka Katyal and Atul Mishra. 2014. A comparative study of load balancing algorithms in cloud computing environment. *International Journal of Distributed and Cloud Computing* 1, 2 (2014), 5–14.

Parmeet Kaur and Shikha Mehta. 2017. Resource provisioning and work flow scheduling in clouds using augmented shuffled frog leaping algorithm. *Journal of Parallel and Distributed Computing* 101 (2017), 41–50.

Rajwinder Kaur and Pawan Luthra. 2012. Load balancing in cloud computing. In *Proceedings of the International Conference on Recent Trends in Information, Telecommunication and Computing (ITC'12)*. Citeseer, 374–381.

Sina Keshvadi and Behnam Faghih. 2016. A multi-agent based load balancing system in IaaS cloud environment. *International Robotics & Automation Journal* 1, 1 (2016), 1–6.

Abdul Nasir Khan, Mat Kiah, Samee Khan, and Sajjad Madani. 2013. Towards secure mobile cloud computing: A survey. *Future Generation Computer Systems* 29, 5 (2013), 1278–1299.

Leili Mohammad Khanli, Seyad Naser Razavi, and Nima Jafari Navimipour. 2008. LGR: The new genetic based scheduler for grid computing systems. In *Proceedings of the IEEE International Conference on Computational Intelligence for Modelling Control & Automation*. IEEE, 639–644.

Md. Tanzim Khorshed, A. B. M. Shawkat Ali, and Saleh A. Wasimi. 2012. A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing. *Future Generation Computer Systems* 28, 6 (2012), 833–851.

Somayeh Kianpisheh, Nasrolah Moghadam Charkari, and Mehdi Kargahi. 2016. Ant colony based constrained workflow scheduling for heterogeneous computing systems. *Cluster Computing* 19, 3 (2016), 1053–1070.

Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.

Barbara Kitchenham, O. Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen Linkman. 2009. Systematic literature reviews in software engineering–a systematic literature review. *Information and Software Technology* 51, 1 (2009), 7–15.

T. Kokilavani and D. I. George Amalarethinam. 2011. Load balanced min-min algorithm for static meta-task scheduling in grid computing. *International Journal of Computer Applications* 20, 2 (2011), 43–49.

Eetu Kupiainen, Mika Mantyla, and Juha Itkonen. 2015. Using metrics in agile and lean software development–a systematic literature review of industrial studies. *Information and Software Technology* 62 (2015), 143–163.

Jiayin Li, Meikang Qiu, Zhong Ming, Gang Quan, Xiao Qin, and Zonghua Gu. 2012. Online optimization for scheduling preemptable tasks on IaaS cloud systems. *Journal of Parallel and Distributed Computing* 72, 5 (2012), 666–677.

Yu Liu, Changjie Zhang, Bo Li, and Jianwei Niu. 2015. DeMS: A hybrid scheme of task scheduling and load balancing in computing clusters. *Journal of Network and Computer Applications* 83 (2015), 213–220.

Flavio Lombardi and Roberto Di Pietro. 2011. Secure virtualization for cloud computing. *Journal of Network and Computer Applications* 34, 4 (2011), 1113–1122.

Radha Ramani Malladi. 2015. An approach to load balancing in cloud computing. *International Journal of Innovative Research in Science, Engineering and Technology (Online)* (2015), 2319–8753.

Yingchi Mao, Daoning Ren, and Xi Chen. 2013. Adaptive load balancing algorithm based on prediction model in cloud computing. In *Proceedings of the 2nd International Conference on Innovative Computing and Cloud Computing (ICCC'13)*. ACM, New York, NY, Article 165, 165–170 pages.

Dan Marinescu, Ashkan Paya, John Morrison, and Stephen Olariu. 2017. An approach for scaling cloud resource management. *Cluster Computing* 20, 1 (2017), 909–924.

Viktor Mauch, Marcel Kunze, and Marius Hillenbrand. 2013. High performance cloud computing. *Future Generation Computer Systems* 29, 6 (2013), 1408–1416.

Carlos Perez Miguel, Alexander Mendiburu, and Jose Miguel Alonso. 2015. Modeling the availability of Cassandra. *Journal of Parallel and Distributed Computing* 86 (2015), 29–44.

Alireza Sadeghi Milani and Nima Jafari. 2016. Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends. *Journal of Network and Computer Applications* 71 (2016), 86–98.

Alireza Sadeghi Milani and Nima Jafari Navimipour. 2016. Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends. *Journal of Network and Computer Applications* 71 (2016), 86–98.

Nader Mohamed, Jameela Al-Jaroodi, and Abdulla Eid. 2013. A dual-direction technique for fast file downloads with dynamic load balancing in the Cloud. *Journal of Network and Computer Applications* 36, 4 (2013), 1116–1130.

Rupam Mukhopadhyay, Dibyajyoti Ghosh, and Nandini Mukherjee. 2010. A study on the application of existing load balancing algorithms for large, dynamic, heterogeneous distributed systems. In *Proceedings of the 9th International Conference on Software Engineering, Parallel and Distributed Systems*. World Scientific and Engineering Academy and Society (WSEAS), 238–243.

Ranesh Kumar Naha and Mohamed Othman. 2016. Cost-aware service brokering and performance sentient load balancing algorithms in the cloud. *Journal of Network and Computer Applications* 75 (2016), 47–57. DOI: https://doi.org/10.1016/j.jnca.2016.08.018

Nima Jafari Navimipour. 2015. A formal approach for the specification and verification of a trustworthy human resource discovery mechanism in the expert cloud. *Expert Systems with Applications* 42, 15 (2015), 6112–6131.

Nima Jafari Navimipour and Yeganeh Charband. 2016. Knowledge sharing mechanisms and techniques in project teams: Literature review, classification, and current trends. *Computers in Human Behavior* 62 (2016), 730–742. DOI: https://doi.org/10.1016/j.chb.2016.05.003

Nima Jafari Navimipour and Farnaz Sharifi Milani. 2015. A comprehensive study of the resource discovery techniques in Peer-to-Peer networks. *Peer-to-Peer Networking and Applications* 8, 3 (2015), 474–492.

Giovanni Neglia, Matteo Sereno, and Giuseppe Bianchi. 2016. Geographical load balancing across green datacenters: A mean field analysis. *ACM SIGMETRICS Performance Evaluation Review* 44, 2 (2016), 64–69.

Kumar Nishant, Pratik Sharma, Vishal Krishna, Chhavi Gupta, Kuwar Pratap Singh, Ravi Rastogi, et al. 2012. Load balancing of nodes in cloud using ant colony optimization. In *Proceedings of the 14th International Conference on Computer Modelling and Simulation (UKSim)*. IEEE, 3–8.

Klaithem Al Nuaimi, Nader Mohamed, Mariam Al Nuaimi, and Jameela Al Jaroodi. 2012. A survey of load balancing in cloud computing: Challenges and algorithms. In *IEEE 2nd Symposium on Network Cloud Computing and Applications (NCCA'12)*. IEEE, 137–142.

Alberto Nunez, Jose Luis Vazquez-Poletti, Agustin C. Caminero, Jesus Carretero, and Ignacio Martin Llorente. 2011. Design of a new cloud computing simulation platform. In *Proceedings of International Conference on Computational Science and its Applications*. Springer, 582–593.

Simon Ostermann, Kassian Plankensteiner, Radu Prodan, and Thomas Fahringer. 2010. GroudSim: An event-based simulation framework for computational grids and clouds. In *Proceedings of European Conference on Parallel Processing*. Springer, 305–313.

Elina Pacini, Cristian Mateos, and Carlos Garcia. 2015. Balancing throughput and response time in online scientific Clouds via ant colony optimization. *Advances in Engineering Software* 84 (2015), 31–47. DOI: https://doi.org/10.1016/j.advengsoft.2015.01.005

Nusrat Pasha, Amit Agarwal, and Ravi Rastogi. 2014. Round robin approach for VM load balancing algorithm in cloud computing environment. *International Journal of Advanced Research in Computer Science and Software Engineering* 4, 5 (2014), 34–39.

Zdzislaw Pawlak. 2001. Rough sets and decision algorithms. *Lecture Notes in Computer Science* Vol. 2005 (2001), 30–45. DOI: https://doi.org/10.1007/3-540-45554-X_3

Florin Pop, Ciprian Dobre, Valentin Cristea, Nik Bessis, Fatos Xhafa, and Leonard Barolli. 2015. Deadline scheduling for aperiodic tasks in inter-Cloud environments: A new approach to resource management. *The Journal of Supercomputing* 71, 5 (2015), 1754–1765.

Roopali Punj and Rakesh Kumar. 2018. Technological aspects of WBANs for health monitoring: A comprehensive review. *Wireless Networks* (2018), 1–33.

Fahimeh Ramezani, Jie Lu, and Farookh Khadeer Hussain. 2014. Task-based system load balancing in cloud computing using particle swarm optimization. *International Journal of Parallel Programming* 42, 5 (2014), 739–754. DOI: https://doi.org/10.1007/s10766-013-0275-4

Martin Randles, David Lamb, and A. Taleb-Bendiab. 2010. A comparative study into distributed load balancing algorithms for cloud computing. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA*. IEEE, 551–556.

Soumya Ray and Ajanta De Sarkar. 2012. Execution analysis of load balancing algorithms in cloud computing environment. *International Journal on Cloud Computing: Services and Architecture* 2, 5 (2012), 1–13.

Maria A. Rodriguez and Rajkumar Buyya. 2017. Scheduling dynamic workloads in multi-tenant scientific workflow as a service platforms. *Future Generation Computer Systems* 79, 2 (2017), 739–750.

Mohsen Amini Salehi, Jay Smith, Anthony A. Maciejewski, Howard Jay Siegel, Edwin K. P. Chong, Jonathan Apodaca, Luis D. Briceno, Timothy Renner, Vladimir Shestak, Joshua Ladd, et al. 2016. Stochastic-based robust dynamic resource allocation for independent tasks in a heterogeneous computing system. *Journal of Parallel and Distributed Computing* 97 (2016), 96–111.

Haiying Shen, Lei Yu, Liuhua Chen, and Zhuozhao Li. 2016. Goodbye to fixed bandwidth reservation: Job scheduling with elastic bandwidth reservation in clouds. In *Proceedings of the IEEE International Conference on Cloud Computing Technology and Science (CloudCom'16)*. IEEE, 1–8.

Jianhui Shi, Chunlei Meng, and Lingli Ma. 2011. The strategy of distributed load balancing based on hybrid scheduling. In *4th International Joint Conference on Computational Sciences and Optimization (CSO'11)*. IEEE, 268–271.

Aarti Singh, Dimple Juneja, and Manisha Malhotra. 2015. Autonomous agent based load balancing algorithm in cloud computing. *Procedia Computer Science* 45 (2015), 832–841.

Priyanka Singh, Palak Baaga, and Saurabh Gupta. 2016. Assorted load-balancing algorithms in cloud computing: A survey. *International Journal of Computer Applications* 143, 7 (2016), 1–8.

Poonam Singh, Maitreyee Dutta, and Naveen Aggarwal. 2017. A review of task scheduling based on meta-heuristics approach in cloud computing. *Knowledge and Information Systems* (2017), 1–51.

Sukhpal Singh and Inderveer Chana. 2015. QoS-aware autonomic resource management in cloud computing: A systematic review. *Computer Surveys* 48, 3, Article 42 (2015), 42 pages. DOI: https://doi.org/10.1145/2843889

Erica Sousa, Fernando Lins, Eduardo Tavares, Paulo Cunha, and Paulo Maciel. 2015. A modeling approach for cloud infrastructure planning considering dependability and cost requirements. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 4 (2015), 549–558.

Subashini Subashini and Veeraruna Kavitha. 2011. A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications* 34, 1 (2011), 1–11.

M. Suresh, S. Karthik, and B. Santosh Kumar. 2014. A load balancing model in public cloud using ANFIS and GSO. In *Proceedings of the International Conference on Intelligent Computing Applications (ICICA'14)*. IEEE, 85–89.

L. Tasquier. 2015. Agent based load-balancer for multi-cloud environments. *Columbia International Publication Journal of Cloud Computing Research* 1, 1 (2015), 35–49.

Technavio. 2017. *Top 10 Cloud Computing Service Providers in 2017*. Retrieved February 20, 2018 from https://www.technavio.com/blog/top-10-cloud-computing-service-providers-2017.

William Voorsluys, James Broberg, and Rajkumar Buyya. 2011. Introduction to cloud computing. *Cloud computing: Principles and Paradigms* 87 (2011), 1–44.

Wei Jen Wang, Yue Shan Chang, Win Tsung Lo, and Yi Kang Lee. 2013. Adaptive scheduling for parallel tasks with QoS satisfaction for hybrid cloud environments. *The Journal of Supercomputing* 66, 2 (2013), 783–811.

Zhenhua Wang, Haopeng Chen, Ying Fu, Delin Liu, and Yunmeng Ban. 2015. Workload balancing and adaptive resource management for the swift storage system on cloud. *Future Generation Computer Systems* 51 (2015), 120–131.

Bhathiya Wickremasinghe, Rodrigo N. Calheiros, and Rajkumar Buyya. 2010. Cloudanalyst: A Cloudsim-based visual modeller for analysing cloud computing environments and applications. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, 446–452.

Tin Yu Wu, Wei Tsong Lee, Yu San Lin, Yih Sin Lin, Hung Lin Chan, and Jhih Siang Huang. 2012. Dynamic load balancing mechanism based on cloud storage. In *Proceedings of the IEEE International Conference on Computing, Communications and Applications*. IEEE, 102–106.

Zhilong Wu, Sheng Xing, Shubin Cai, Zhijiao Xiao, and Zhong Ming. 2016. A genetic-ant-colony hybrid algorithm for task scheduling in cloud system. In *Proceedings of the International Conference on Smart Computing and Communication*. Springer, 183–193.

Zhen Xiao, Weijia Song, and Qi Chen. 2013. Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE Transactions on Parallel and Distributed Systems* 24, 6 (2013), 1107–1117. DOI: https://doi.org/10.1109/TPDS.2012.283

Yu Xin, Zhi Qiang Xie, and Jing Yang. 2017. A load balance oriented cost efficient scheduling method for parallel tasks. *Journal of Network and Computer Applications* 81 (2017), 37–46.

Moona Yakhchi, Seyed Mohssen Ghafari, Shahpar Yakhchi, Mahdi Fazeli, and Ahmad Patooghi. 2015. Proposing a load balancing method based on cuckoo optimization algorithm for energy management in cloud computing infrastructures. In *Proceedings of the 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO'15)*. IEEE, 1–5.

Jixiang Yang, Ling Ling, and Haibin Liu. 2016. A hierarchical load balancing strategy considering communication delay overhead for large distributed computing systems. *Mathematical Problems in Engineering* 2016 (2016), 1–9.

Faouzia Zegrari, Abdellah Idrissi, and Hajar Rehioui. 2016. Resource allocation with efficient load balancing in cloud environment. In *Proceedings of International Conference on Big Data and Advanced Wireless Technologies (BDAW'16).* ACM, New York, NY, Article 46, 7 pages. DOI: https://doi.org/10.1145/3010089.3010131

Lingfang Zeng, Bharadwaj Veeravalli, and Albert Y. Zomaya. 2015. An integrated task computation and data management scheduling strategy for workflow applications in cloud environments. *Journal of Network and Computer Applications* 50 (2015), 39–48.

Qi Zhang, Lu Cheng, and Raouf Boutaba. 2010. Cloud computing: State-of-the-art and research challenges. *Journal of Internet Services and Applications* 1, 1 (2010), 7–18.

Jia Zhao, Kun Yang, Xiaohui Wei, Yan Ding, Liang Hu, and Gaochao Xu. 2016. A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment. *IEEE Transactions on Parallel and Distributed Systems* 27, 2 (2016), 305–316.

Dimitrios Zissis and Dimitrios Lekkas. 2012. Addressing cloud computing security issues. *Future Generation Computer Systems* 28, 3 (2012), 583–592.

Received December 2017; revised August 2018; accepted August 2018