Afzal and Kavitha *Journal of Cloud Computing: Advances, Systems and Applications* (2019) 8:22
https://doi.org/10.1186/s13677-019-0146-7

Journal of Cloud Computing: Advances, Systems and Applications

REVIEW **Open Access**

Load balancing in cloud computing – A hierarchical taxonomical classification

Shahbaz Afzal* and G. Kavitha

Abstract

Load unbalancing problem is a multi-variant, multi-constraint problem that degrades performance and efficiency of computing resources. Load balancing techniques cater the solution for load unbalancing situation for two undesirable facets- overloading and under-loading. In contempt of the importance of load balancing techniques to the best of our knowledge, there is no comprehensive, extensive, systematic and hierarchical classification about the existing load balancing techniques. Further, the factors that cause load unbalancing problem are neither studied nor considered in the literature. This paper presents a detailed encyclopedic review about the load balancing techniques. The advantages and limitations of existing methods are highlighted with crucial challenges being addressed so as to develop efficient load balancing algorithms in future. The paper also suggests new insights towards load balancing in cloud computing.

Keywords: Cloud computing, Taxonomy, Classification, Cloud service consumer, Cloud service provider, Quality of service, Load unbalancing, Load balancing

Introduction

Cloud Computing is an internet based network technology that shared a rapid growth in the advances of communication technology by providing service to customers of various requirements with the aid of online computing resources. It has provisions of both hardware and software applications along with software development platforms and testing tools as resources [1, 2]. Such a resource delivery is accomplished with the help of services. While as the former comes under category of Infrastructure as a service (IaaS) cloud, the latter two comes under headings of Software as a service (SaaS) cloud and platform as a service (PaaS) cloud respectively [3]. The cloud computing is an on-demand network enabled computing model that share resources as services billed on pay-as-you-go (PAYG) plan [4]. Some of the giant players in given technology are Amazon, Microsoft, Google, SAP, Oracle, VMware, Sales force, IBM and others [1, 2]. Majority of these cloud providers are high- tech IT organizations. The cloud computing model is viewed under two different headings. The first one is the service delivery model, which defines the type of the service offered by a typical cloud provider. Based on this aspect, there are popularly following three important service models SaaS, PaaS and IaaS [5, 6]. The other aspect of cloud computing model is viewed on its scale of use, affiliation, ownership, size and access. The official 'National Institute of Standards and Technology' (NIST) definition for cloud computing outlines four cloud deployment models namely private, public, community and hybrid clouds [7].

A cloud computing model is efficient if its resources are utilized in best possible way and such an efficient utilization can be achieved by employing and maintaining proper management of cloud resources. Resource management is achieved by adopting robust resource scheduling, allocation and powerful resource scalability techniques. These resources are provided to customers in the form of Virtual Machines (VM) through a process known as virtualization that makes use of an entity (software, hardware or both) known as hypervisor [8]. The greatest advantage of cloud computing is that a single user physical machine is transformed into a multiuser virtual machines [9, 10]. The Cloud Service Provider (CSP) plays a crucial role in service delivery to users and is a complex task with given available virtual resources. While serving user requests, some VMs will get a heavy traffic of user tasks and some will get a lesser traffic. As

* Correspondence: shabazafzal_it_phd_2017@crescent.education
Department of Information Technology, B S Abdur Rahman Crescent Institute of Science and Technology, Chennai 600048, India

a result, the Cloud Service Provider (CSP) is left with unbalanced machines which have a huge gradient of user tasks and resource utilization [11].

The problem of load unbalancing is an undesirable event in the CSP side that degrades the performance and efficacy of the computing resources along with guaranteed Quality of Service (QoS) on agreed Service Level Agreement (SLA) between consumer and provider. Under these circumstances there arises need for load balancing (LB) and is a peculiar topic of research interest among researchers. The load balancing in cloud computing can be done at physical machine level or VM level [2].

A task utilize resources of a VM and when a bunch of tasks arrive at a VM, the resources gets exhausted which means no resource is now available to handle the additional task requests. When such situation arises the VM is said to have entered into an overloaded state. At this point of time, tasks will either suffer from starvation or end up in deadlock with no hope of accomplishing them. Consequently there is necessity to migrate tasks to another resource on other VM. The workload migration process includes three basic steps: load balancing which checks the current load on machine resource, resource discovery which finds another suitable resource and workload migration which moves extra tasks to available resources. These operations are performed by three different units commonly known as load balancer, resource discovery and task migration units respectively.

Load balancing is the process of redistribution of workload in a distributed system like cloud computing ensuring no computing machine is overloaded, under-loaded or idle [12, 13]. Load balancing tries to speed up different constrained parameters like response time, execution time, system stability etc. thereby improving performance of cloud [14, 15]. It is an optimization technique in which task scheduling is an NP hard problem. There are a large number of load balancing approaches proposed by researchers where most of focus has been concerned on task scheduling, task allocation, resource scheduling, resource allocation, and resource management. To the best of our knowledge, we could not find an in-depth and comprehensive literature concerned with factors that cause load unbalancing situation. The survey papers based on load balancing could not provide a proper systematic classification of methods and techniques. The main aim of the paper is to review the existing work along with the advantages and pitfalls in them. A comparison is also made among different existing load balancing techniques and the challenges faced in cloud load balancing. The survey also outlines factors responsible for load unbalancing problem and also suggests methods that can be used in future work. The contributions of this paper are summarized as follows:

I. Explore the factors that cause load unbalancing problem in cloud computing.
II. Provide a systematic overview of the existing approaches in the load balancing process and the way in which these approaches have been used in the cloud technology.
III. Provide the in-depth classification of different load balancing techniques, methods, strategies and algorithms.
IV. Analyze the challenges faced by researchers in developing an efficient load balancing algorithm.

The remaining paper is structured as follows. Section "Load balancing model background" features a brief description about load balancing model in cloud computing. Section "Research methodology" highlights some related works. The research methodology is discussed in section "Research methodology". Section "Proposed classification of load balancing algorithms" proposes taxonomy based classification. The results are evaluated in section "Results and discussion" while section "Discussion on open issues on load balancing in cloud computing" discusses upon open issues in cloud load balancing. Finally section "Conclusion and future work" concludes our work and points out some future directions.

Load balancing model background

In this section a two level load balancing architecture model is presented in imbalanced clouds for achieving best load shedding as shown in Fig. 1 which is a modified architecture given by Gupta et al. [16]. The virtual machine manager and virtual machine monitor are abstracted in this model. The first level load balancing is performed at the Physical Machine (PM) level and the second level is performed at the VM level. Based on this, there are two task migration sets;

1. Intra VM task migration
2. Inter VM task migration

The request generator generates user requests which are user tasks that need computing resources for their execution. Data center controller is in-charge of task management. The load balancer checks which VM to assign for a given user task. The first level load balancer balances the given workload on individual Physical Machines by distributing the workload among its respective associated Virtual Machines. The second level load balancer balances the workload across different Virtual Machines of different Physical Machines.

Activities involved in load balancing

Scheduling and allocating tasks to VMs based on their requirements constitute the cloud computing workload. The load balancing process involves the following activities [2]:

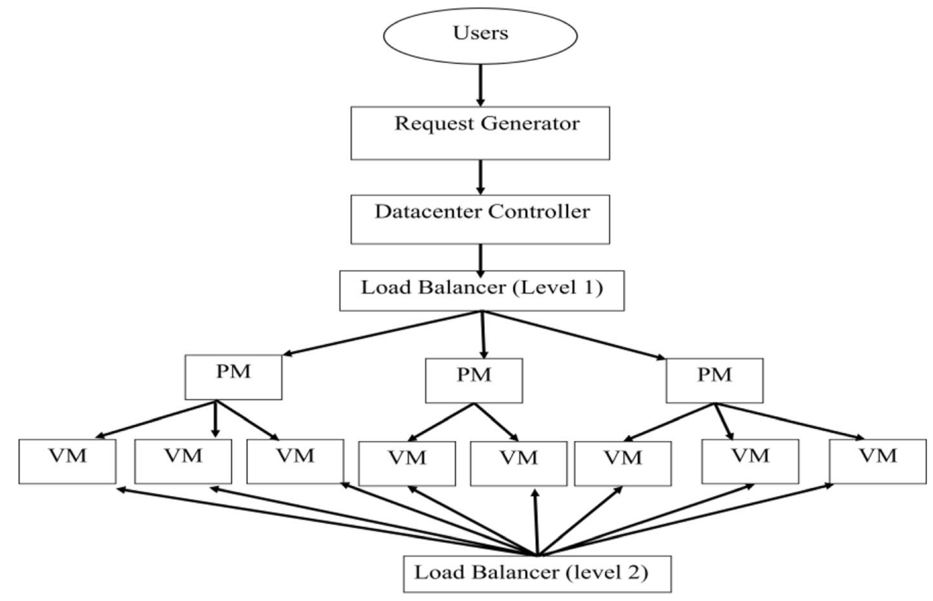

Fig. 1 Two level Load Balancing Architecture

Identification of user task requirements
This phase identifies the resource requirement of the user tasks to be scheduled for execution on a VM.

Identification of resource details of a VM
This checks the status of resource details of a VM. It gives the current resource utilization of VM and the unallocated resources. Based on this phase, the status of VM can be determined as balanced, overloaded or under-loaded with respect to a threshold.

Task scheduling
Once resource details of a VM are identified the tasks are scheduled to appropriate resources on appropriate VMs by a scheduling algorithm.

Resource allocation
The resources are allocated to scheduled tasks for execution. A resource allocation policy is being employed to accomplish this. A large number of scheduling and allocation policies are proposed in literature. While, scheduling is required for speeding up the execution, allocation policy is used for proper resource management and improving resource performance. The strength of the load balancing algorithm is determined by the efficacy of the scheduling algorithm and the allocation policy [17–19].

Migration
Migration is an important phase in load balancing process in cloud and latter is incomplete without the former. Migration is of two kinds in cloud based on entity taken into consideration- VM migration and task migration. VM migration is the movement of a VM from one physical host to another to get rid of the overloading problem and is categorized into types as live VM migration and non live migration. Likewise task migration is the movement of tasks across VMs and is of two types: intra VM task migration and inter VM task migration. A large number of migration approaches has been proposed in literature. An efficient migration technique leads to an efficient load balancing. From the extensive survey it has been concluded that task migration process is more time and cost effective than VM migration and the trend has shifted from VM to task migration [20–24].

Related work
In general a lot of work have been done in the field of cloud computing particularly in scheduling (tasks, VMs and Compute), resource provisioning, resource management, energy management and load balancing etc. However, load balancing has been an eagle's eye among researchers because of its essence in cloud computing between the stakeholders' i.e. Cloud Service Provider and Cloud Service Consumer. Based on analysis of existing review literature one of the reasons presented is absence of proper classification among different approaches. A thorough review about the existing work in literature has been presented in this section.

Ghomi et al. [25] proposed a survey on load balancing algorithm in cloud computing. The authors presented classification on task scheduling and load balancing algorithms in seven different categories that include hadoop- map reduce load balancing, agent based load balancing, natural phenomena based load balancing, application oriented load balancing, general load balancing, network aware load balancing and workflow specific load balancing which in literature fall under two domains based on system state and who initialized the process. From each category, the different algorithms are grouped together and their advantages and limitations are listed. Meanwhile, Milani et al. [26] reviewed existing load balancing techniques, established on the survey; authors grouped existing algorithms into three broad domains as static, dynamic and hybrid. The authors formalized relevant questions towards load balancing and addressed key concern about importance, expectation level of metrics, role and challenges faced in load balancing. A proper search operation was followed in search query to retrieve most relevant content from different publishing sources assisted by Boolean operations in search strings and selection criteria phase was executed with Quality Assessment Checklist (QAC). However the two surveys examined limited QoS metrics in their work that are Response time, Makespan, Scalability, Resource utilization, Migration Time, Throughput and Energy saving leaving behind a gap to consider other important QoS metrics like migration cost, service level violations, degree of balance, task rejection ratio etc. This gap in metric selection for analysis is overcome in this survey.

Kalra and Singh [27] conducted a comparative study of various scheduling algorithms for cloud and grid computing considering five fundamental meta-heuristic methods namely Ant Colony Optimization (ACO), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), League Championship Algorithm (LCA) and BAT algorithm. Besides this a thorough comparison is made among the techniques; however their work is limited to scheduling algorithms for meta-heuristic techniques only. Also the survey concentrates on evolutionary algorithms only and lacks broad classification.

Mesbahi and Rahmani [28] classified load balancing algorithms into three categories: general algorithm based, architectural based and artificial intelligence based, studied the basic requirements and essentials in designing and implementing a desired load balancer for cloud provider. Like other previous studies, this paper considers static and dynamic categorization as broad classification. However, authors suggested key challenges in designing load balancing algorithms. Further, authors made a judgment on the basis of study on algorithms that have properties of being dynamic, distributive and non-cooperative are best.

Kanakala et al. [29] proposed a classification paper on existing load balancing algorithms which are grouped into static and dynamic algorithms like ones discussed in previous studies. They also identified challenges in finding solution for problem of load balancing. Among the challenges are geographical distribution of nodes, migration time, system performance, energy management and security which are long ago listed in the literature. Infact the authors compared existing load balancing algorithms on the basis of certain QoS metrics like throughput, speed, response time, migration time etc. The paper concluded that there is tradeoff among metrics. The limitation of paper is that only eight load balancing algorithms are compared from a vast set of algorithms.

Shah et al. [30] discusses a comprehensive overview with respect to the load balancing algorithms. The different load balancing methods were classified as static and dynamic based on the state of the system, homogeneous and heterogeneous based on VM type uniformity. Performance metrics were also used to classify the load balancing methods. Further, the advantages and disadvantages of each algorithm were discussed. The paper does not address the literature in a systematic manner.

Neghabi et al. [31] presented a well defined, systematic and potential review about load balancing techniques in software defined networks and broadly classified them into deterministic and non-deterministic approaches along with associated metrics being investigated into depth. The study poses some important questions and tries to answer them along dimensions of their significance, metric analysis, role and challenges being faced in load balancing of software defined networks. The study carried out by the authors presents the detailed advantages and limitations of existing literature in communication networks. Further the paper holds a strong foundation and solid correlation among load balancing metrics, despite the fact that it does not go specific to cloud computing domain. Also, the study is based on single level classification rather than hierarchical classification.

From above listed survey papers, it is concluded that already existing survey papers are lacking from a good classification system. A criterion is fixed for classification purpose, but no generalization and specialization characteristics are drawn, which eventually lead to inadequate and insufficient conclusions. Further, existing review articles does not examine some important parameters like algorithmic complexity of load balancing algorithms and also the percentage of occurrence of load balancing metrics in literature. The existing survey papers lacks full description of QoS metric set, most likely new metrics (as migration cost, service level violations, degree of balance, task rejection ratio) should have been introduced in survey. A taxonomy based classification is proposed in this paper to prove its effectiveness over the existing literature. Also, the classification of QoS metrics is proposed in this survey as performance metrics and economic metrics [32]. So, to guide future researchers in developing an efficient, robust, fault tolerant and advanced load balancing algorithm and to give them new insights into future work, a taxonomy based classification system is introduced in this paper. The proposed methodology of classification is based on various characteristics of load balancing algorithms- 'nature of algorithm', 'state of the algorithm', 'trait used for load balancing', 'mode of execution', 'type', 'functionality', and 'technique used by algorithm'.

Research methodology

To go deep into roots of load balancing process as to what causes load unbalancing problem a proper research methodology was followed. The literature survey was conducted in accordance with general research strategy that outlines the way in which load unbalancing problem is undertaken and identifies the methods, theories, algorithms, approaches and paradigms used in it. The load unbalancing problem was studied in accordance with constructive generic framework (CGF) methodology [33] where it is broken down into sub- processes i.e. the factors, variables and parameters that are associated with load balancing. Further the literature study was enhanced by following the research guideline for Systematic Literature Review (SLR) as contemplated by Kitchenham with a special focus on research related to load balancing mechanism in cloud [26, 34]. An SLR is a repeated research method that can be replicated by other researchers to explore more knowledge.

In order to feature the importance of load balancing in cloud computing, a set of questions were framed to address the key issues and challenges in load unbalancing.

Question identification

A set of questions were identified from literature survey that need to be answered before going into the load balancing process. Some of the questions have been answered in literature while others are not. The questions are given as follows:

RQ1: What causes the load unbalancing problem? This question tries to answer why load unbalancing problem happens. This involves identification of factors responsible for cause of load unbalancing. The question cannot be answered until each individual factor is considered and studied with full detail. The load balancing process is incomplete unless knowledge about variables leading to unbalancing is not clear. This is of prime importance and till date no study considered this question. So, this paper presents answer of this question.

RQ2: Why load balancing is the need of hour in cloud computing? This question tries to answer the issues and challenges faced by cloud service providers.

RQ3: Does load balancing consider the evaluation of single objective (single attribute) or multi-objective (multi-attribute) function(s)? This question classifies the existing load balancing algorithms into single objective and multi-objective approaches.

RQ4: What is the time complexity of load balancing algorithm? This question answers the amount of time taken by load balancing algorithm to complete the load balancing process. Algorithm complexity is not taken as a standard for classifying the LB algorithms in the existing literature. The algorithm should run with real time algorithm complexity to be of practical use.

This section explores the causes for load unbalancing problem in IaaS clouds and tries to answers RQ1. The following factors are the causes for load unbalancing problem in IaaS clouds.

- The dynamic nature of user tasks.
- The unpredictable and probabilistic traffic flow to a cloud provider.
- Lack of robust, accurate and efficient mapper and generator function to map the tasks to the appropriate resources.
- The scheduling process itself is an NP hard problem.
- The heterogeneous nature of user tasks demanding varying resource requirements.
- The uneven and non- uniform distribution of tasks across computing resources along with their dependencies also contributes to load unbalancing situation.

Load balancing is a promising solution to load unbalancing problem that arise due to circumstances discussed in this section. This section answers RQ2 with the importance of load balancing in cloud computing. The load balancing algorithm has to enhance response time, cost of execution, execution time, throughput, fault tolerance, migration time, degree of balance, makespan, resource utilization and scalability. At the same time to reduce the resource wastage, migration cost, power consumption, energy consumption, carbon emission and SLA violations. The degradation of values of these factors leads to poor Quality of Service (QOS) to CSC and drop in economy in the form of profit to CSP. So keeping in view QOS and economy, it has become a big challenge for CSPs to provide QOS according to guaranteed SLA. However, to improve performance and economic metrics in one go is still a milestone for researchers to conclude the load balancing as NP hard problem like scheduling. This is because as we try to improve one specific metric the associated metrics begin to diminish and bottleneck persists, thus declaring load unbalancing as a multi- constrained multi-objective problem.

The section discusses classification of load balancing approaches as single objective and multi-objective on the basis of number of objective functions solved by a particular algorithm and tries to answer RQ3. RQ3 is also an elucidation of the RQ2 to consider load unbalancing as a multi-objective problem. Till date there exists no perfect load balancing algorithm in literature that takes into accounts all of the metrics in a single algorithm. Different researchers propose single objective algorithm to speed up a single metric while others tried to improve more than one metric at a time. The limitation of single objective approaches is that introduction of these in load balancing process altogether would result in huge complexity of architectural design and may become impractical for use. So prime attention is shifted from single objective to multi-objective approaches. Table 3 reviews different existing approaches based on single objective or multi-objective function(s) solved by a particular approach.

The RQ4 tries to answer time complexity of the algorithm being used in load balancing process and should be considered as a benchmark to determine performance of a load balancing algorithm. However, as a matter of concern we could not find enough literature determining the algorithmic complexity of an approach being used in the process. Out of top 35 studies conducted in this research only 7 studies considers the algorithmic complexity in their work which accounts to only 20% and the figure may drop as we increase the search space.

Milani et al. [26] identified the three main primary questions in the existing literature and justified them in their work. The questions were formalized as follows;

1. What is the consequence on load balancing with the growth of cloud users? The authors pointed out that from 2010 to 2015 there is a momentous rise in research papers on the scope of load balancing that follows a positive exponential curve and in our work find it more increasing in 2016, 2017 and 2018. This shows the importance of load balancing in the cloud computing with increasing number of users.
2. What is the capability of present load balancing approaches to meet the primary load balancing metrics? The question was answered and validated through the argument that dynamic load balancing algorithms are more practical, robust, efficient and fault tolerant than static ones.
3. What are the problems, issues, challenges and solutions identified in load balancing for future trends? The limitations and advantages of the existing approaches were listed and based on that challenges faced by researchers were discussed.

Data collection search process

The data collection search process includes papers gathered from reputed sources, journals and publications from five most authentic and potential databases that are scientifically and technically peer- reviewed: IEEE Xplore Digital Library, Science Direct, ACM Digital Library, Springer and Elsevier. The search was organized in June 2018 with data collected from 2010 to June 2018. The data source consist of review and survey papers, journal papers and conference papers excluding book chapters. A well organized search process was adopted to retrieve relevant data started with fundamental terms to advanced ones. The search strings were framed for source databases with inclusion and exclusion criteria similar to the one used by [26]. Search keywords were formed along with their synonyms to increase the search space. Initially, the basic terms and keywords were used in search query processing like “Load Balancing in Cloud Computing”, “Workload Distribution in Cloud Computing”, “Resource Distribution in Cloud”, “Task scheduling”, “Migration process in Cloud Computing”, “Resource utilization in Cloud”, “Resource allocation policies in Cloud”, “Load scheduling in Cloud” and much more. Later on the advanced terms were used following by integration of basic keywords in query operation assisted by Boolean operations as “Boolean OR” and “Boolean AND” [26, 35] to narrow down the search space for relevant data. As an illustration the following keywords were integrated “Resource allocation AND Task scheduling”, “Task Migration” “Task scheduling” and “Resource utilization”. Later on the advanced search operation was equipped to collect most reliable papers for data collection, like the use of Inspec controlled and non- controlled keywords were applied followed by advanced filters. As an example the following advanced search operation was applied in the IEEE Xplore. “Advanced keyword/phrase” option which include two sub options “metadata only” and “full text and metadata” each using three Boolean operations “AND”, “OR” “NOT”. Similarly, command search and citation search options were also used.

Proposed classification of load balancing algorithms

In this section load balancing algorithms are classified based on various criteria. A top down approach is proposed and followed in classification process. The limitation of existing review papers is that there is no proper and significant hierarchical taxonomical classification of load balancing algorithms which makes it quite difficult to identify where a particular algorithm holds its place in taxonomy. The various criteria used for classification purpose include ‘nature of algorithm’, ‘state of algorithm’, ‘trait used for load balancing’, ‘type of load balancing’, and ‘technique used in load balancing ’. For the first time in literature an in-depth analysis of the LB algorithms have been achieved in this work which the previous studies were lacking. Based on nature of algorithm, the load balancing algorithms are either proactive or reactive. This is first broad categorization we placed in taxonomy and which till date have not been shown in any of the literature studies. Based on state of system, the LB algorithms are static, dynamic or hybrid. On the basis of trait used in load balancing, LB algorithms are classified as scheduling and allocation algorithms. On the basis of type of LB algorithms they are grouped as VM LB algorithms, CPU LB algorithms, Task LB algorithms, Server LB algorithms, Network LB algorithms and Normal Cloud Load balancing algorithms. On the basis of functionality, the load balancing methods are grouped as hardware load balancing; elastic load balancing and latter is further grouped into network load balancing, application load balancing and classic load balancing. Based on the technique, Load balancing algorithms are classified as; Machine learning, Evolutionary, Nature Inspired, mathematical derived algorithms, and swarm based techniques.

Nature of the algorithm

The first categorization of load balancing algorithms in this work has been done on the basis of nature of algorithm. On the basis of this classification, LB algorithms are classified as proactive based approaches and reactive based approaches. However in other fields of technology particularly in the communication and networking for mobile adhoc networks (MANETS), the nature of the communication routing protocols has been extensively studied under these two variants [36].

A proactive based LB algorithmic technique is an approach to algorithmic design which takes into consideration action by causing change and not only reacting to that change when it happens. It is intended to yield a good outcome to avoid a problem in advance rather than waiting until there is a problem. Proactive behavior aims at identification and exploitation of opportunities and in taking preemptory action against potential problems and threats. The limitation of existing approaches is that a limited number of proactive approaches have been used and that too in a traditional manner with no novel concepts. Table 1 depicts proactive approaches in existing LB approaches. Polepally et al. [37] proposed dragonfly optimization and constraint measure-based LB approach in cloud computing by distributing uniform load across VMs with minimal power consumption. Xiao, et al. [38] proposed game theory based fairness-aware LB algorithm to minimize expected response time while maintain fairness. The Nash equilibrium point of the game corresponds to load balancing at optimal level.

The reactive based approaches act in response to a situation rather than controlling it. In reactive based approach of load balancing the problem of load unbalancing is solved as it arises and after which the consequences are visible. Most of load balancing algorithms fall under this category. The main flaw that has been analyzed from literature of existing works on load balancing is that load unbalancing problem is left to happen and then researchers propose some approaches to tackle that problem by optimizing some load balancing parameter(s) [32] as given in Fig. 2. Table 2 discusses the reactive approaches in the existing LB approaches e.g. Adhikari et al. [39] proposed a heuristic-based scheduling and load balancing algorithm for IaaS cloud to minimize task completion time, makespan, waiting time, and increase resource utilization. Proactive approaches are more effective than reactive approaches as the former tries to avoid the problem in advance while in latter the solution is provided after the problem occurs.

State of the algorithm

On the basis of state information of system that an algorithm relies on, LB algorithms are widely classified as static, dynamic and hybrid. From existing literature survey, it is evident that this is most widely used classification system for LB algorithms. Majority of work on comparative studies on load balancing begin the algorithmic taxonomy by placing this category on top of taxonomy. In static load balancing, traffic load is segregated uniformly across the servers. This is done by algorithm having the prior knowledge about system resources and task requirements. The static LB algorithm schedules tasks to VM for execution at compile time. The advantage of static algorithm is their less complexity but they suffer from a fatal bottleneck of being unable to move tasks during execution in progress to another machine for load balancing. Static algorithms do not consider current state of system and requires advance knowledge about machines and tasks like task resource requirements, communication time, processing power of nodes, memory, storage, bandwidth capacity and so on. The main drawback of static LB algorithm is that migration process is not possible during execution of tasks and hence is not suitable approach for distributed system like cloud where system state changes dynamically.

Further on the basis of mode of execution of tasks, dynamic algorithms are grouped as offline mode also called as batch mode and online mode or live mode as shown in Fig. 3. In batch mode, the task is allocated only at some predefined instances where as in online mode the user task is mapped to a VM as soon it enters the scheduler. Dynamic load balancing algorithms are comparatively complex algorithms in contrast with their counterparts that handle incoming traffic flow at run time and can change state of a running task at any point of time. Dynamic load balancing takes into consideration the current state of system and has capacity to deal with unpredictable processing load. The advantage of dynamic load balancing is that tasks can move dynamically from an overloaded machine to under-loaded one but are much complex in

Table 1 Proactive approaches of Cloud load balancing in existing literature

Reference	Algorithm Used	Trait Used	Type of Load Balancing	Technique involved	Algorithm Complexity	Advantages	Disadvantages
[40]	Conventional Non Classical	Task Scheduling	Task LB	Heuristic (Classical, Deterministic)	Not Specified	Capable of handling heavy workloads within predefined deadline.	Tasks whose execution time is more than defined deadline are rejected.
						Provides enhanced elasticity.	Thresholds for defining overloaded and under loaded VMs are set arbitrarily without formulating equation for them.
						Minimize makespan with improved task acceptance ratio.	
						Minimize task rejection ratio	The experimental are run on Cloudsim using space shared policy only and not time shared policy.
						Perform automatic scaling of resources	
[41]	Full Set algorithm and Column generation algorithm	VM scheduling	VM LB	Optimization (Classical, Deterministic, LP)	$[O(2)^N - n\ O(2^k)/2]$	Load balancing is performed among minimum number of VMs	Algorithm evaluates only single objective function.
						Improved resource utilization	The experiments are run on C++ programs
						Resource over provisioning is avoided	
						The algorithm runs in real-time scale with simple complexity.	
[37]	Dragonfly optimization and constraint measure-based load balancing	Task Scheduling	Task LB	Optimization (Swarm Based)	Not Specified	Load balancing is performed with less power consumption	Cannot handle tasks beyond threshold limit.
							Task rejection ratio is high
[38]	Fairness Aware Algorithm	Resource Scheduling	CPU LB	Optimization (non cooperative game theory based)	Not Specified	Optimal Lb is achieved at Nash equilibrium point.	High task execution time
						Minimize expected response time	
[42]	Honey Bee Behaviour	Task Scheduling	Task LB	Optimization (Swarm Based)	Not Specified	Low response time.	Low scalability
						Low makespan	
[43]	ACO	Task Scheduling	Task LB	Optimization (Swarm Based)	Not Specified	Less makespan	Tasks are mutually independent
						Measures degree of imbalance among VMs	Memory intensive tasks are not taken
[44]	Agent based Nature Inspired Algorithm	Resource Scheduling	Resource LB	Metaheuristic	Not Specified	High scalability	Execution cost not considered
						Less response time	Service level violations not considered
						Improved resource utilization	Task rejection rate not considered
[45]	Non- Classical	Resource Scheduling	Resource LB	Heuristic	Not Specified	High fault tolerance	High response time
						Less overhead	High execution time
							High makespan
[46]	Weighted Round Robin	Resource Scheduling	Server LB	Heuristic	O(1)	Good resource utilization	response time not chosen
						Enhanced throughput	degree of balance not chosen

Table 1 Proactive approaches of Cloud load balancing in existing literature *(Continued)*

Reference	Algorithm Used	Trait Used	Type of Load Balancing	Technique involved	Algorithm Complexity	Advantages	Disadvantages
[47]	Nature Inspired GA	Task and Resource Scheduling	Task LB	Optimization	O(1)	Less overhead High fault tolerance Efficient resource utilization Less resource wastage Small energy consumption Less SLV Improved degree of balance	energy efficiency not chosen Priority based Less scalability Less fault tolerance

nature and much complicated to design compared to static LB algorithms.

However dynamic LB algorithms are much efficient in terms of performance, accuracy and functionality. Static load balancing algorithms work smoothly if nodes have small load variations but could not operate in varying load environments. Figure 3 shows the load balancing taxonomy on the basis of nature and state of algorithm.

Trait used for load balancing

The algorithms in this category are classified as scheduling and allocation algorithms. The allocation and scheduling algorithms in cloud are classified based upon current state of

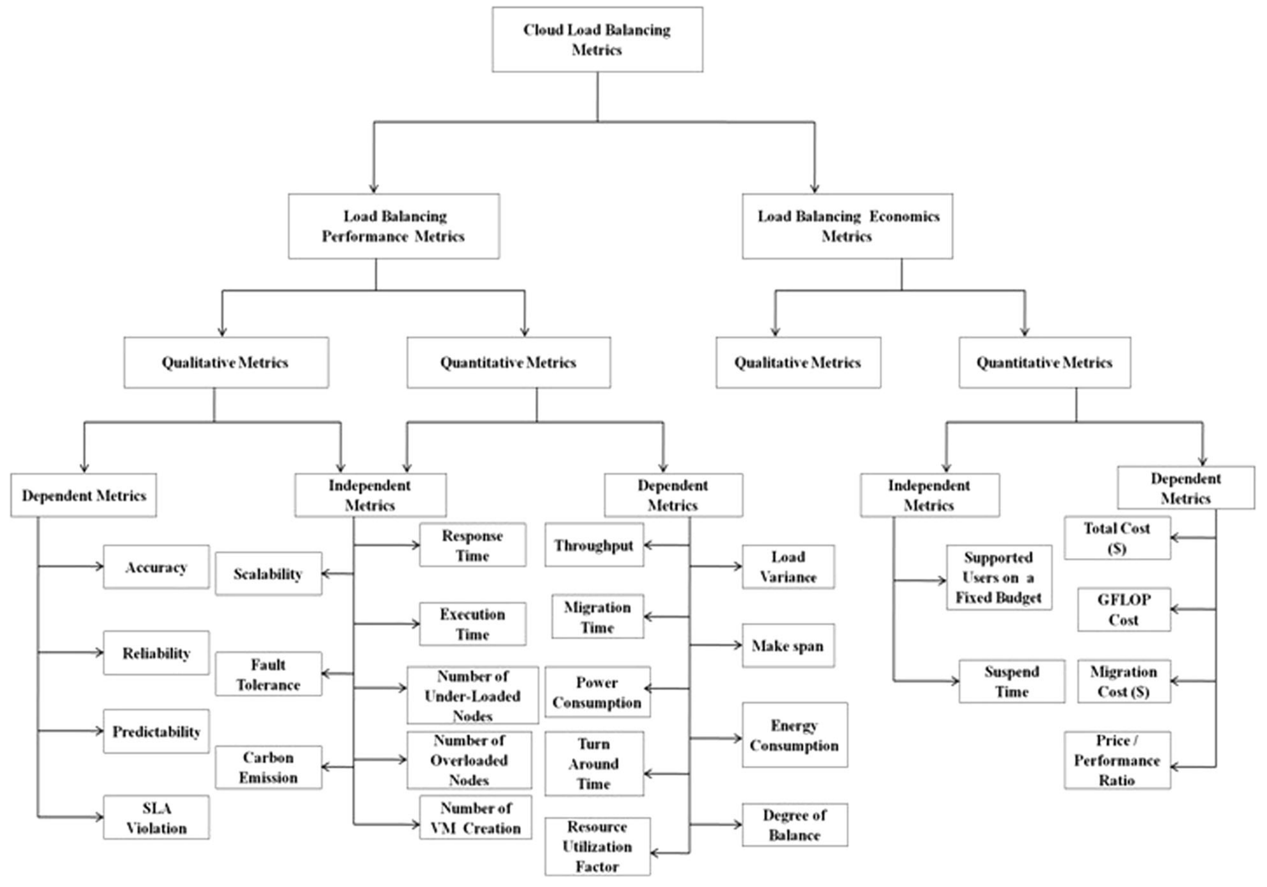

Fig. 2 Load Balancing Metrics [32]

Table 2 Reactive approaches of cloud load balancing in existing literature

Reference	Algorithm Used	State of Algorithm	Trait Used	Type of Load Balancing	Technique Involved	Algorithm Complexity	Advantages	Disadvantages
[48]	Conventional non classical Algorithm	Dynamic	Task Scheduling	Task LB	Non Classical, Deterministic	Not Specified	Better makespan	Task deadline not considered
							Better resource utilization	SLV not considered
							Less waiting time	Less fault tolerant
							Less execution time	Less energy efficient
[39]	Classical and Linear Programming	Dynamic	Task Scheduling	Task LB	Optimization (Linear programming Based)	Not Specified	Better makespan Better resource utilization	Reduced quality of service
[49]	GA and Min-Min	Hybrid	Task Scheduling	Task LB	Heuristic (Evolutionary)	O(m) and O (mn)	Better scalability	Less resource utilization
							Less response time	High SLV
							Small execution cost	Less degree of balance
[50]	BFO+ Lamarack Evolutionary	Hybrid	Resource Scheduling	CPU LB	Optimization	Not Specified	Low VM downtime, execution time	Low scalability and throughput
							Less transfer time	Low resource utilization
[51]	PSO	Dynamic	Task Scheduling	Task LB	Optimization	Not Specified	Low energy consumption	Low scalability
							High resource utilization	Low fault tolerance
								Small degree of balance
								Less makespan
								High SLV
[52]	GA	Dynamic	VM Scheduling	VM LB	Metaheuristic	Not Specified	Less response time	Low throughput
							Less makespan	Low scalability
							Less task rejection ratio	Small degree of balance
								Small resource utilization
[53]	GA	Dynamic	Task Scheduling	Task/VM LB	Optimization	$G = O\{n1 + (c \times k) + (n2 + 1)\ (m + m + m)\}$	High degree of balance	Low scalability
							Less makespan	low energy efficiency
							Less execution time	low fault tolerance
							Less task rejection ratio	
[54]	ACO and PSO	Dynamic	VM Scheduling	VM LB	Metaheuristic	O(n^2MAI)	low response time	low throughput
							low execution time	low degree of balance
								high SLV
								low resource utilization

Table 2 Reactive approaches of cloud load balancing in existing literature *(Continued)*

Reference	Algorithm Used	State of Algorithm	Trait Used	Type of Load Balancing	Technique Involved	Algorithm Complexity	Advantages	Disadvantages
[55]	GA and GEL	Hybrid	Task Scheduling	VM LB	Optimization	Not Specified	high scalability	low degree of balance
							high fault tolerance	high SLV
							low overhead	low resource utilization
							low migration time and power consumption	high TRR
[56]	Honey Bee Algorithm	Dynamic	Task Scheduling	Task LB	Optimization	Not Specified	low response time	low throughput and scalability
							low execution time	low degree of balance
							low execution cost	low resource utilization
[57]	Non Classical	Dynamic	Resource Scheduling	Resource LB	Heuristic	Not Specified	High throughput	Low SLV
							High scalability	Low resource utilization
							Low response time	High task rejection ratio
							Low execution time	Low degree of balance
								High migration time
[58]	Non Classical	Dynamic	VM Scheduling	VM LB	Optimization	Not Specified	Low migration time	Low throughput
							High degree of balance	Low makespan
							Low response time	High SLV
								Low resource utilization
								Low scalability
[59]	BAT Algorithm	Dynamic	Resource/ Task Scheduling	Resource/ Task LB	Optimization	Not Specified	Less execution time	High makespan
							Low execution cost	Low throughput
								Energy inefficient
								Low resource utilization
[60]	Non Classical	Dynamic	VM Scheduling	VM LB	Optimization	Not Specified	Less response time	Low scalability
							Low execution cost	High SLV
								Low degree of balance
[61]	Simulated Annealing	Dynamic	Task Scheduling	Task LB	Optimization	Not Specified	High throughput	Low fault tolerance
							High scalability	Energy inefficient
							Low overhead	High SLV
							Less makespan	
							High resource	

Table 2 Reactive approaches of cloud load balancing in existing literature *(Continued)*

Reference	Algorithm Used	State of Algorithm	Trait Used	Type of Load Balancing	Technique Involved	Algorithm Complexity	Advantages	Disadvantages
							utilization	
[62]	Round Robin	Static	VM Scheduling	VM LB	Heuristic	Not Specified	High Fault tolerance	Less scalability
							Small overhead	High SLV
							Less migration time	Low task rejection ratio.
							Good resource utilization	
[63]	Round Robin	Dynamic	Resource Scheduling	Resource LB	Heuristic	Not Specified	High Fault tolerance	Less scalability
							Less migration time	High SLV
							Good resource utilization	Low task rejection ratio
[64]	Non Classical	Static	Resource Scheduling	Resource LB	Optimization	Not Specified	Less Response time	Low throughput and scalability
							Low execution cost	Low resource utilization
								Low degree of balance
[65]	Active Monitoring	Dynamic	VM Scheduling	VM LB	Heuristic	Not Specified	Less response time	Low throughput
							Less execution time	Low scalability
							Less execution cost	Low degree of balance
								Low resource utilization
[66]	Active Monitoring	Dynamic	VM/Task Scheduling	VM/Task LB	Heuristic	Not Specified	High scalability	Low throughput
							Less response time	Low fault tolerance
							High resource utilization	High makespan
								High SLV
[67]	Active Monitoring	Dynamic	Resource Scheduling	Resource LB	Heuristic	Not Specified	Low overhead	Low throughput
							Less makespan	Power inefficient
							High resource utilization	High SLV
[68]	Joint use of min-min and max-min	Static	Task Scheduling	Task LB	Optimization	Not Specified	High degree of balance	Low scalability
							Low makespan	Low fault tolerance
							Low execution time	High SLV
							High resource utilization	
[69]	Min-min	Static	Task/ Resource Scheduling	Task/ Resource LB	Optimization	Not Specified	Low makespan	Low throughput and scalability
							Low response time	High SLV and task rejection ratio
							High resource	Power

Table 2 Reactive approaches of cloud load balancing in existing literature *(Continued)*

Reference	Algorithm Used	State of Algorithm	Trait Used	Type of Load Balancing	Technique Involved	Algorithm Complexity	Advantages	Disadvantages
							utilization	inefficient
[70]	Max-Min	Static	Task Scheduling	Task LB	Optimization	O $(mn)^2$	High throughput and scalability	Low resource utilization
							Low fault tolerance	Low degree of balance
							Low overhead	High makespan
[71]	Round Robin	Dynamic	Task Scheduling	Task LB	Heuristic	Not Specified	Low makespan	Low fault tolerance
							Low power consumption	Low degree of balance
							Low SLV	Low resource utilization

VM and accordingly can be static or dynamic. The allocation and scheduling policies play a vital role in resource management and performance monitoring of cloud, which in turn has a good impact on QoS delivery to user. The scheduling policies are decomposed into three subsequent activities task scheduling, resource scheduling and VM scheduling; likewise the allocation policies are decomposed as task allocation, resource allocation and VM allocation respectively.

Task scheduling is the method of assigning user tasks to relevant computing resources for execution while resource scheduling is the process of planning, managing and monitoring computing resources for task execution. VM scheduling is the process of creating, destroying and managing VMs within a physical host apart from managing the VMs during migration process across the hosts. The task allocation is the act of allocating task to a resource on which it is supposed to execute. Resource allocation is the act of allocating a resource to a task for its completion. Task allocation and resource allocation are inversion of each other. VM allocation is the allocation of virtual machine to a user or a set of users. Figure 4 shows the load balancing algorithms on the basis of trait being used.

Functionality

On the basis of functionality, load balancers are classified as hardware load balancer and elastic load balancer as depicted in Fig. 5. Hardware load balancers are concerned with the distribution of workload at hardware level i.e. memory, storage and CPU. Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses. It can handle varying load of user application traffic in a single availability zone or across multiple availability zones. Elastic Load Balancing offers three types of load balancers that all feature high availability, automatic scaling, and robust security necessary to make user applications fault tolerant. Application Load Balancer operates at the request level (layer 7) routing traffic to targets - EC2 instances, containers and IP addresses based on the content of the request. Ideal for advanced load balancing of HTTP and HTTPS traffic,

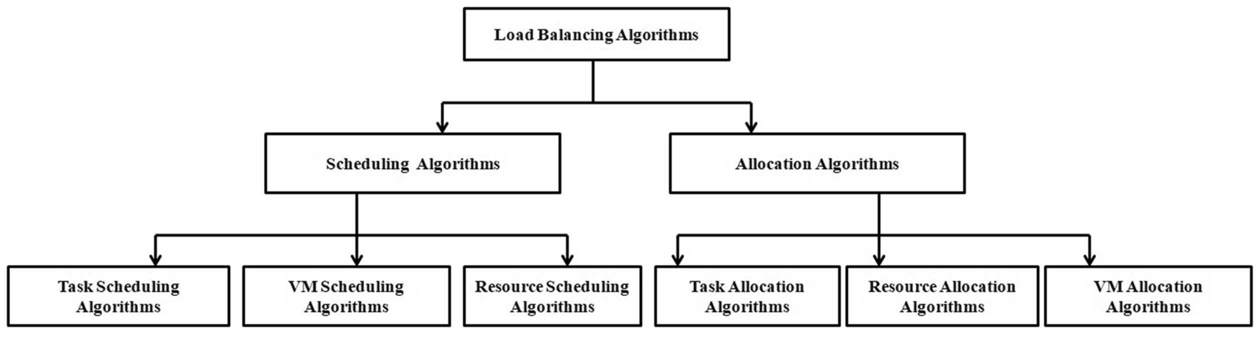
Load Balancing Algorithms
Scheduling Algorithms
Allocation Algorithms
Task Scheduling Algorithms
VM Scheduling Algorithms
Resource Scheduling Algorithms
Task Allocation Algorithms
Resource Allocation Algorithms
VM Allocation Algorithms

Fig. 3 Load balancing algorithms on the basis of nature and state of the system

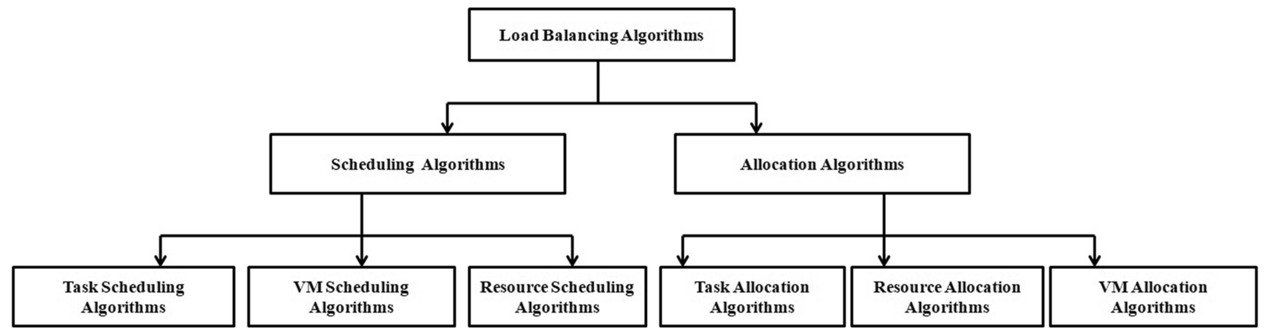
Load Balancing Algorithms
Scheduling Algorithms
Allocation Algorithms
Task Scheduling Algorithms
VM Scheduling Algorithms
Resource Scheduling Algorithms
Task Allocation Algorithms
Resource Allocation Algorithms
VM Allocation Algorithms

Fig. 4 Load balancing on the basis of trait used

Application Load Balancer provides advanced request routing targeted at delivery of modern application architectures, including micro-services and container-based applications. Application Load Balancer simplifies and improves the security of your application, by ensuring that the latest SSL/TLS ciphers and protocols are used at all times. Network load balancers are implemented at the transport layer of the OSI model. It has ability to handle millions of requests per second. The network load balancing is popularly used by Microsoft azure and AWS in deployment model. The network load balancing feature allows traffic distribution among servers using the TCP/IP internet protocol. Classic Load Balancer provides basic load balancing across multiple Amazon EC2 instances and operates at both the request level and connection level. Classic Load Balancer is intended for applications that were built within the EC2-Classic network.

Type of load balancing

On the basis of type, LB algorithms are classified as VM LB, CPU LB, task LB, server LB, network LB and normal cloud LB as shown in Fig. 5. VM load balancing is the process of redistribution of VMs from overloaded nodes to under loaded nodes and was first introduced as a new inbox feature in windows server 2016 that allows optimize node utilization in a failure cluster. VM load balancing identifies over committed nodes and redistributes VMs from those nodes to under committed nodes. VMs are live migrated from a node exceeding threshold to a newly added node in failure cluster. VM load balancing is achieved through VM migration process. CPU load balancing is the process of limiting the load on a CPU within its threshold limit. Task Load balancing is the act of distribution of tasks across the VMs from overloaded machines to under loaded machines. Server LB is proper distribution of total incoming load in a datacenter or a server farm across the servers. Network LB is concerned with management of incoming traffic without use of complex protocols.

Technique used in load balancing

On the basis of the technique used, load balancing algorithms are classified as heuristics and meta-heuristics techniques, and optimization techniques.

A heuristic approach is an approach to problem solving accounting a practical method or methodology guaranteed

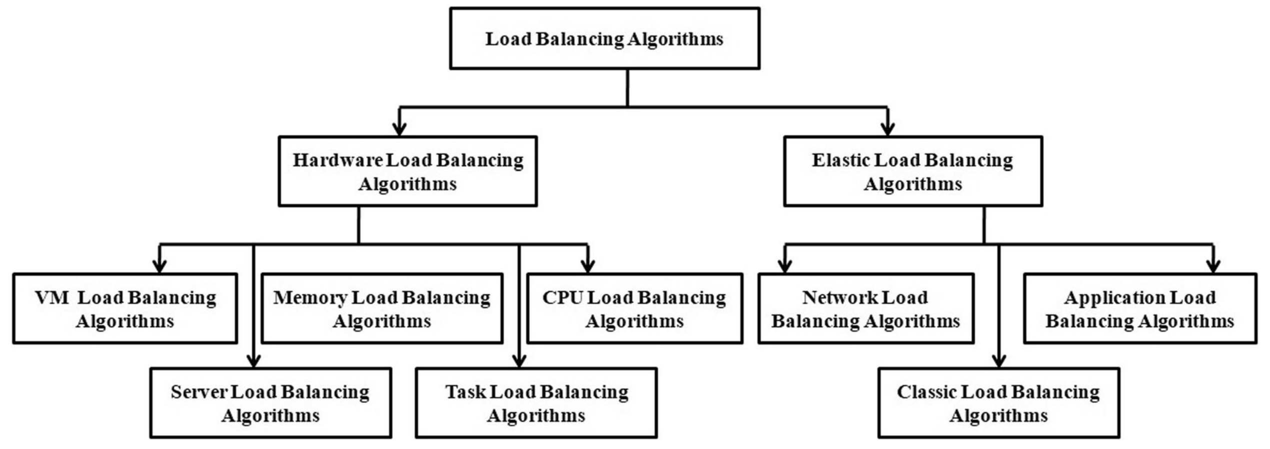
Load Balancing Algorithms
Hardware Load Balancing Algorithms
Elastic Load Balancing Algorithms
VM Load Balancing Algorithms
Memory Load Balancing Algorithms
CPU Load Balancing Algorithms
Network Load Balancing Algorithms
Application Load Balancing Algorithms
Server Load Balancing Algorithms
Task Load Balancing Algorithms
Classic Load Balancing Algorithms

Fig. 5 Load balancing on the basis of functionality and type

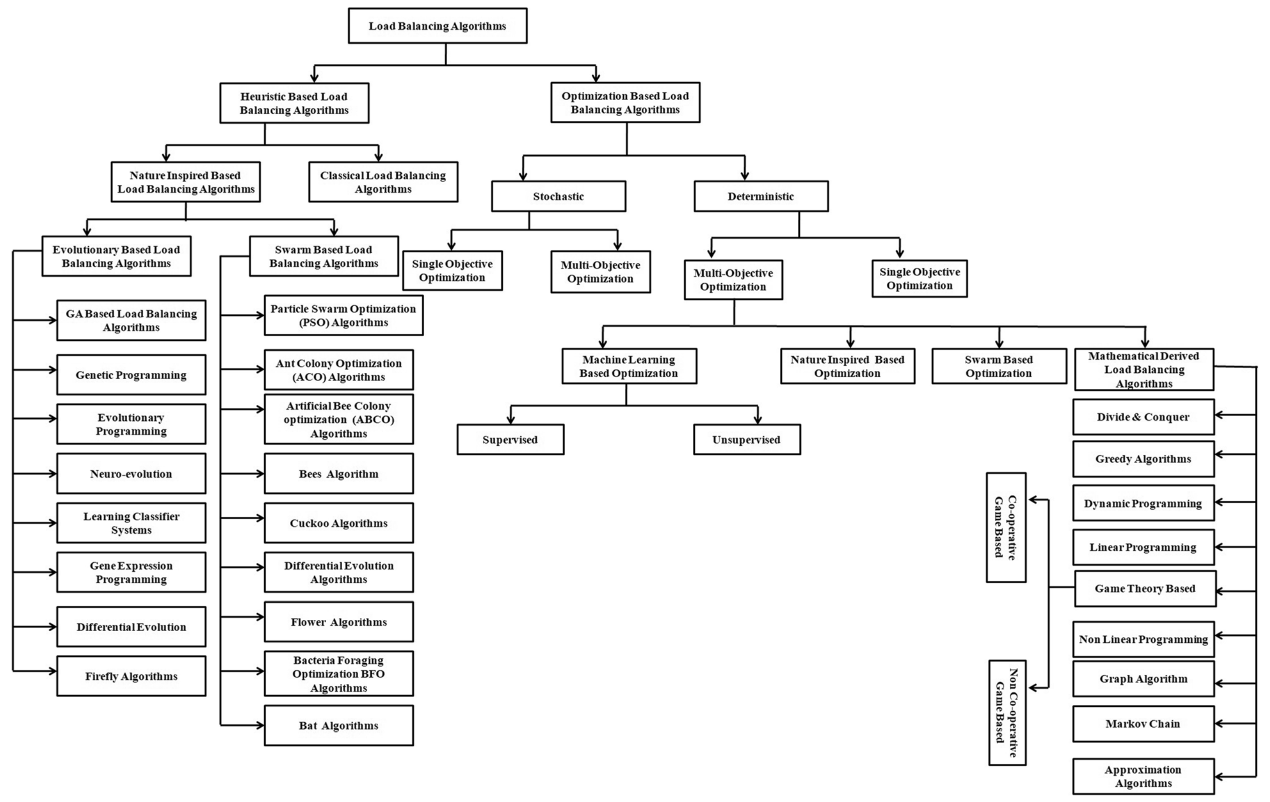
Load Balancing Algorithms
Heuristic Based Load Balancing Algorithms
Optimization Based Load Balancing Algorithms
Nature Inspired Based Load Balancing Algorithms
Classical Load Balancing Algorithms
Stochastic
Deterministic
Evolutionary Based Load Balancing Algorithms
Swarm Based Load Balancing Algorithms
Single Objective Optimization
Multi-Objective Optimization
Multi-Objective Optimization
Single Objective Optimization
GA Based Load Balancing Algorithms
Genetic Programming
Evolutionary Programming
Neuro-evolution
Learning Classifier Systems
Gene Expression Programming
Differential Evolution
Firefly Algorithms
Particle Swarm Optimization (PSO) Algorithms
Ant Colony Optimization (ACO) Algorithms
Artificial Bee Colony optimization (ABCO) Algorithms
Bees Algorithm
Cuckoo Algorithms
Differential Evolution Algorithms
Flower Algorithms
Bacteria Foraging Optimization BFO Algorithms
Bat Algorithms
Machine Learning Based Optimization
Nature Inspired Based Optimization
Swarm Based Optimization
Mathematical Derived Load Balancing Algorithms
Supervised
Unsupervised
Divide & Conquer
Greedy Algorithms
Dynamic Programming
Linear Programming
Game Theory Based
Co-operative Game Based
Non Co-operative Game Based
Non Linear Programming
Graph Algorithm
Markov Chain
Approximation Algorithms

Fig. 6 Load balancing algorithms on the basis of technique

not be optimal, perfect, logical or rationale but sufficient to reach an immediate goal. Finding an optimal solution may be impossible or impractical particularly to load balancing which is a NP hard problem and heuristics play an important role to gear up the process of finding a decent solution. Heuristic methods are designed following strategies derived from previous experience with similar problem sets. Heuristics play a crucial role in load balancing process to sort up various issues faced by CSPs. A lot of research work has been carried out with heuristic and meta-heuristic approaches in cloud load balancing and as such we have classified the heuristic and meta-heuristic methods into nature inspired algorithms and classical algorithms. The nature inspired algorithms are sub divided into evolutionary based algorithms and swarm based algorithms.

Optimization techniques are used to find optimal solutions of a problem. Optimization techniques in cloud load balancing are broadly classified as classical and non classical optimization techniques. These algorithms can be either stochastic or deterministic. A further classification classifies the optimization techniques into constrained based and non constrained based algorithms and these can further be either a single objective or multi-criteria optimization. The multi- criteria optimization is further classified as multi-attribute and multi-objective optimization. The multi-objective algorithms may be either machine learning based, nature inspired based, swarm based or mathematical derived based load balancing algorithms. Figure 6 shows the load balancing algorithms on the basis of technique used.

Table 1 and Table 2 respectively list the different characteristics of proactive and reactive approaches in related literature along various dimensions. The strength and weakness of each approach is also reflected in Table 1 and Table 2. Table 3 depicts the different approaches under investigation as single objective and multi-objective. Table 3 also highlights the implementation platform, tool and simulating environment under which a particular approach was studied and investigated. Finally Table 4 presents the essential load balancing metrics analyzed in the existing approaches.

Results and discussion

This section outlines the results achieved from comparative analysis of different load balancing approaches in cloud computing. Figure 7(a) shows the percentage of various scheduling types in proactive based load balancing approaches. It is clear that task scheduling and resource scheduling each with 45.45% contribution are more often considered in proactive based approaches with less attention towards VM scheduling which

Table 3 Number of objective function (s) and implementation platform used in existing literature

Approach	Objective	Platform
[40]	Multi-Objective	CloudSim
[41]	Single-Objective	C++
[37]	Multi-Objective	CloudSim
[38]	Single-Objective	SP2
		Scalable power parallel 2 system
[42]	Multi-Objective	CloudSim
[43]	Multi-Objective	CloudSim
[44]	Multi-Objective	Java
[45]	Single-Objective	Real Time Trace Implementation
[46]	Single-Objective	MS Visual Studio, C# with My SQl Server 2005 databases
[47]	Multi-Objective	Matlab
[48]	Multi-Objective	CloudSim
[39]	Multi-Objective	C++ and Matlab
[49]	Multi-Objective	CloudSim
[50]	Single-Objective	CloudSim
[51]	Multi-Objective	CloudSim
[52]	Single-Objective	VMware ESX server
[53]	Single-Objective	Cloud Analyst
[54]	Multi-Objective	C
[55]	Single-Objective	Cloud Analyst
[56]	Single-Objective	CloudSim and Workflowsim
[57]	Multi-Objective	PHP & MySQl
[58]	Single-Objective	Cloud Analyst
[59]	Multi-Objective	Matlab
[60]	Single-Objective	Cloud Analyst
[61]	Single-Objective	Cloud Analyst
[62]	Single-Objective	CloudSim
[63]	Single-Objective	CloudSim
[64]	Multi-Objective	Euculaptus
[65]	Single-Objective	Cloud Analyst
[66]	Multi-Objective	NA
[67]	Multi-Objective	Cloud Analyst
[68]	Multi-Objective	Theoretical Analysis
[69]	Multi-Objective	Matlab
[70]	Multi-Objective	C++
[71]	Multi-Objective	CloudSim

contribute 9.09%. From Fig. 7(b) it is evident that most of the reactive approaches in existing literature have been studied under task scheduling which amount to 51.85%, followed by VM scheduling which contribute to 25.93% and resource scheduling which contribute to 22.22% respectively. Figure 8 describes the percentage of research articles on cloud load balancing defining the algorithmic complexity. It is calculated that 80% of research articles did not considered algorithmic complexity in their work while only 20% define it in their work. It is analyzed from Fig. 9(a) that proactive approaches are always dynamic in nature while Fig. 9(b) depicts that most of the reactive approaches fall under dynamic state of algorithm which contribute to 68%, followed by static

Table 4 Load balancing metrics in existing literature

Reference	Throughput	Scalability	Fault Tolerance	Overhead	Migration Time	Degree of Balance	Makespan	Response Time	Execution Time	Execution Cost	Power Consumption	Service Level Violation	Resource Utilization	Migration Cost	Task Rejection Ratio	Waiting Time
[48]							*			*			*			
[40]		*					*								*	
[41]		*		*									*			
[39]							*		*				*			*
[37]		*												*		
[49]							*		*				*			
[50]		*						*		*						
[38]						*		*	*							
[42]	*					*	*		*							
[51]					*				*	*						
[52]											*		*			
[53]							*	*								
[54]						*	*		*						*	
[43]						*	*		*							
[55]								*	*							
[44]	*	*						*		*			*			*
[56]		*	*	*	*		*				*					
[45]	*	*	*	*												
[57]								*	*	*						
[58]	*	*						*	*	*						
[59]					*	*		*								
[60]									*	*						
[61]								*		*						
[46]	*	*	*	*									*			
[62]	*	*		*				*					*			
[63]	*		*	*	*								*			
[64]	*		*	*	*			*			*		*			
[65]								*		*						
[66]	*							*	*	*						

Table 4 Load balancing metrics in existing literature *(Continued)*

Reference	Throughput	Scalability	Fault Tolerance	Overhead	Migration Time	Degree of Balance	Makespan	Response Time	Execution Time	Execution Cost	Power Consumption	Service Level Violation	Resource Utilization	Migration Cost	Task Rejection Ratio	Waiting Time
[67]		*						*					*			
[68]				*			*	*					*			*
[69]						*	*		*				*			
[70]							*	*	*				*			
[71]	*	*	*	*				*	*							
[47]							*				*	*				

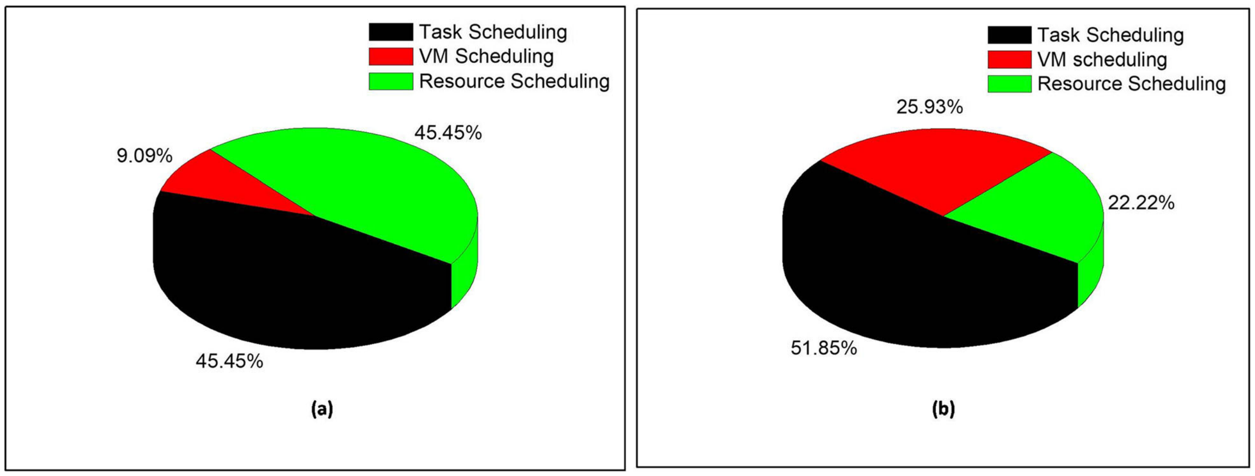

Fig. 7 Percentage based on scheduling trait (task scheduling, VM scheduling, and resource scheduling) in proactive and reactive approaches

algorithm which contribute 20% and hybrid algorithm which represent 12%. It is evident from Fig. 10(a) that 60% of proactive approaches are multi-objective while 40% are single objective approaches. Likewise 56% of reactive approaches are multi-objective while 44% are single objective approaches as depicted in Fig. 10(b). Figure 11 displays the testing environment on which a particular approach was run to evaluate the performance metrics. It is clear that CloudSim simulator is extensively used for conducting simulation experiment constituting 33.33% of experimental implementation followed by Cloud Analyst simulator with 19.44% of experimental implementation. C and C++, Matlab implementation of load balancing approaches amount to 11.11% each respectively while others constitute 19.44%. The real time implementation of cloud load balancing approaches is very less and constitutes only 5.56%. Figure 12 depicts the percentage of LB metrics in the existing approaches where response time, execution time, resource utilization, makespan, scalability and execution cost are most widely discussed each with 13.39%, 11.81%, 11.02%, 9.45%, 9.45% and 8.66% respectively.

Discussion on open issues on load balancing in cloud computing

The review presented in this article addresses some important issues that had not been taken with good consideration in existing survey literature neither in technical literature and which the cloud load balancing demands rigorously. Thus, we discuss some open research in this section.

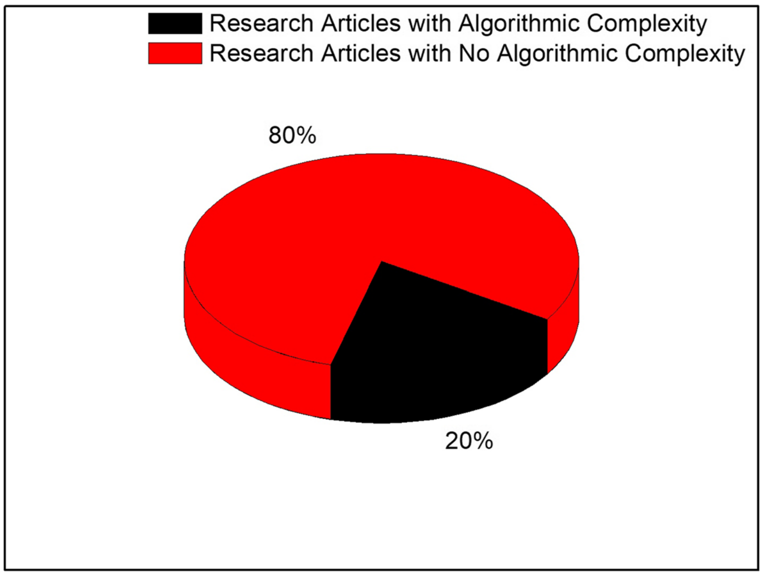

Fig. 8 Percentage of research articles designating algorithmic complexity

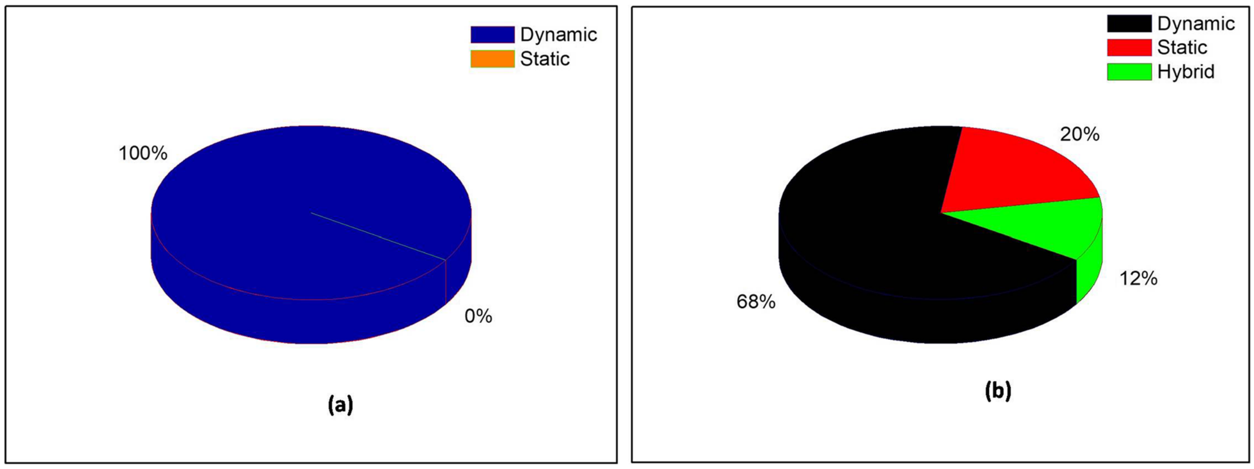

Fig. 9 Percentage based on state of algorithm in proactive (dynamic and static)and reactive (dynamic, static, and hybrid) approaches

Complexity of an algorithm is a pivotal element in determining the performance of any load balancing algorithm. Out of 35 potential technical articles considered in this study, it is found that only 7 articles define the corresponding algorithmic complexity which amounts to 20% and also 28 articles does not define the algorithmic complexity which amounts to 80%. Therefore, it is observed that majority of the works does not include algorithmic complexity and hence for future researchers it is suggested that algorithm complexity should be made a benchmark for developing a new load balancing approach with improved practicality.

A reactive approach of load balancing always features migration in particular task migration. Migration of tasks always incurs some cost that is called as migration cost. From the study it is evident that less literature in cloud load balancing focuses on migration cost apart from Service Level Violations, Task Rejection ratio and power consumption. This can be considered as important direction for future researchers in development of reactive approaches with minimum migration cost.

Further, from the study carried out in this work, it is investigated that majority of the works primarily focus on certain metrics and avoids other main metrics. Out of 16 different metrics collected in this study it is revealed that most of existing works on cloud load balancing features 6 metrics as key parameters for evaluation that are response time (13.39%), execution time (11.81%), resource utilization (11.02%), makespan (9.45%), Scalability (9.45%) and execution cost (8.66%) respectively as depicted in Fig. 12. while the remaining 10 metrics account to only 36.22% and they are Throughput (7.87%), Overhead (7.09%), Fault Tolerance

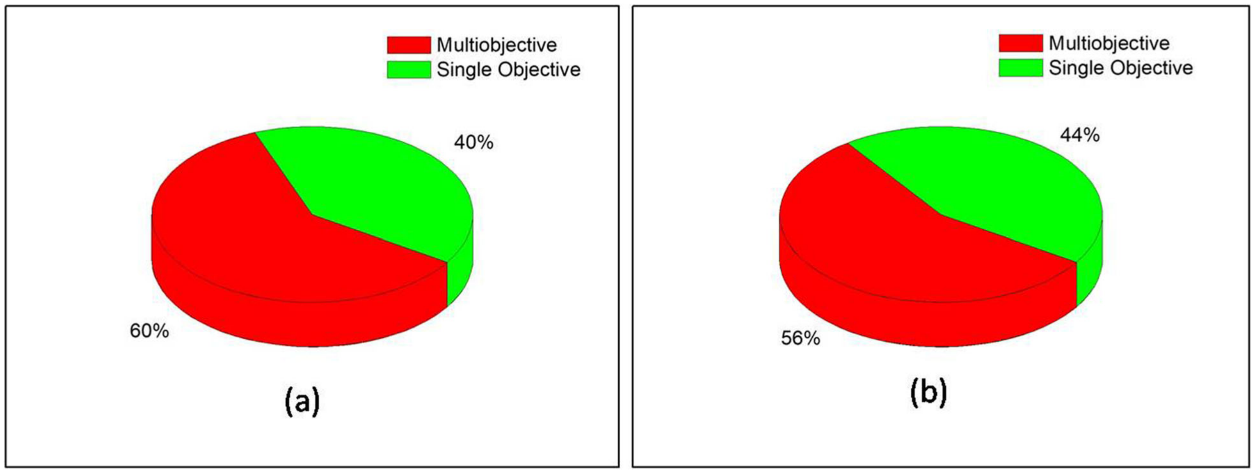

Fig. 10 Percentage of multi-objective and single objective algorithms in proactive and reactive approaches

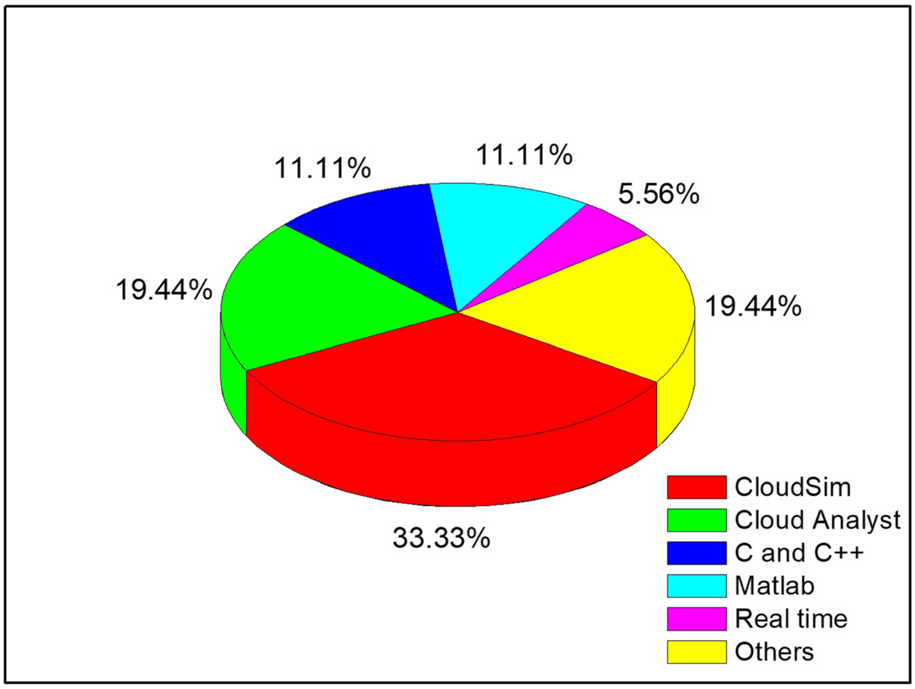

Fig. 11 Experimental platforms for cloud load balancing approaches

(4.72%), Degree of balance (4.72%), Migration Time (3.93%), Power consumption (3.14%), Waiting time (2.36%), Task Rejection Ratio (1.50%), and Service Level Violation (0.78%). considering these metrics in future works is also one of the insights for future researchers.

Conclusion and future work

The work presents a comparative study on load balancing approaches in reviewed articles. The problem of load unbalancing in cloud computing was discussed along with driving factors that lead to this problem. An

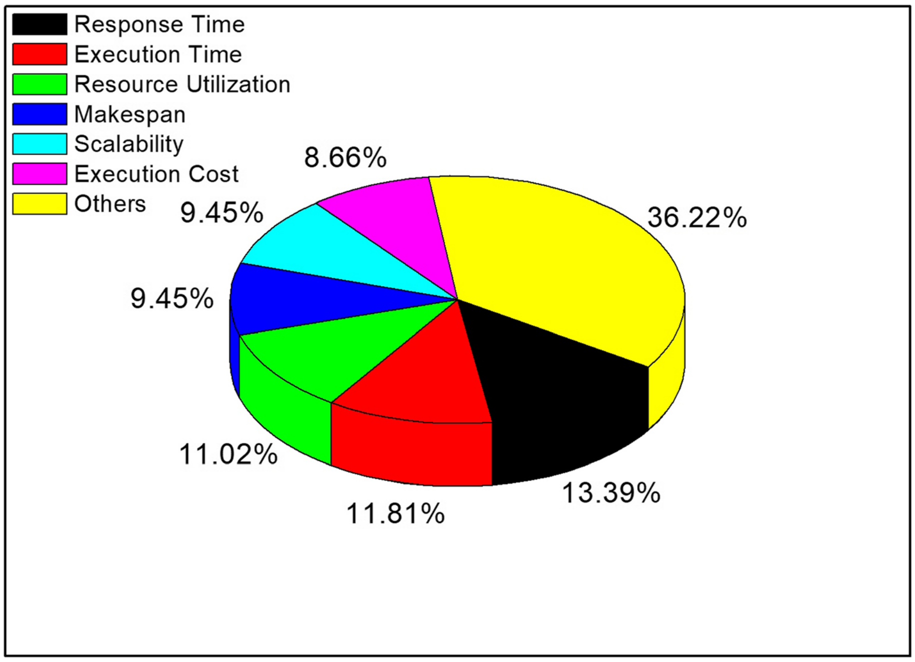

Fig. 12 Percentage of LB metrics in Existing approaches

abstracted load balancing model was briefly discussed together with activities involved in load balancing process. A proper research methodology was followed in which the problem was studied in guidelines with Constructive Generic Framework (CGF) further reinforced by Systematic Literature Review (SLR) methodology. We framed a set of problem related questions and discussed them in the work. The data collected for this study had been gathered from five reputed potential databases that include IEEE Xplore digital library, Science Direct, ACM digital Library, Springer and Elsevier. The data search process was assisted by different tools and advanced filter options. The data was collected for the period from 2010 to June 2018. A multilevel taxonomy based classification was proposed in this work where the classification process is done on five criteria. The most important criteria used in this work is "Nature of Algorithm". Based on this criteria we classified 35 articles into two broad categories- 10 of them are proactive and 25 of them are reactive in nature. The statistics showed that proactive approaches are 100% dynamic while reactive approaches need not be dynamic. We also generalized that all proactive approaches are dynamic but all dynamic approaches may not be proactive Also the study revealed that task scheduling had been given much importance both in proactive and reactive approaches contributing 45% and 51.85% respectively.

The challenges of the load balancing algorithms are explored in this work in order to suggest more efficient load balancing methods in future. Majority of the reviewed articles had not considered significant and fundamental QoS metrics for investigation. Some of the essential QoS metrics are not discussed in reviewed articles in full depth e.g. migration time, migration cost, power consumption, service level violation, task rejection ratio and degree of balance. Further our study revealed that algorithm complexity is not given much attention in determining the performance of load balancing algorithm and as such 80% of the works does not consider it for evaluation of performance. Also majority of existing load balancing approaches have been implemented on simulator platforms which overall constitute 94.44%. Real time implementation of load balancing is very less (5.56) and should be encouraged in future works.

From the review conducted during this research process, it is concluded that there are a lot of issues still open in load balancing process which can be bridged in future by applying an efficient and sophisticated load balancing algorithm most importantly along dimensions of additional QoS metrics and algorithm complexity evaluation. The survey also presents some algorithms in taxonomy which can guide the future researchers to deal with load unbalancing problem effectively like nature inspired algorithms, machine learning and mathematical derived algorithms (Markov chain, game theory based).

Abbreviations

IaaS: Infrastructure as a service; SaaS: Software as a service; PaaS: Platform as a service; PAYG: Pay-as-you-go; NIST: National Institute of Standards and Technology; VM: Virtual Machine; CSP: Cloud Service Provider; CSC: Cloud Service Consumer; QoS: Quality of Service; SLA: Service Level Agreement; ALO: Ant Lion optimizer; CGF: Constructive generic framework; SLR: Systematic literature review; LB: Load Balancing; PM: Physical Machine; MANETS: Mobile adhoc networks; VPC: Virtual Private Cloud

Acknowledgements

The authors are grateful to the editor and anonymous referees for their valuable comments and suggestions. Only the authors are responsible for the views expressed and mistakes made.

Authors' contributions

Design the study: SA. Collected the data from different sources: SA and GK. Analysis and interpretation of data: SA. Drafting of Manuscript: GK. All authors read and approved the final manuscript.

Funding

NA

Availability of data and materials

The data has been gathered from research papers and articles that are mentioned in Table 1, Table 2, Table 3 and Table 4.

Competing interests

The authors declare that there is no competing interest.

Received: 21 January 2019 Accepted: 25 November 2019
Published online: 23 December 2019

References

1. Pradhan P, Behera PK, Ray BNB (2016) Modified round Robin algorithm for resource allocation in cloud computing. Proced Comp Sci 85:878–890
2. Mishra SK, Sahoo B, Parida PP (2018) Load balancing in cloud computing: a big picture. J King Saud Univ Comp Infor Sci:1–32
3. Reddy VK, Rao BT, Reddy LSS (2011) Research issues in cloud computing. Glob J Comp Sci Technol 11(11):70–76
4. Bohn RB, Messina J, Liu F, Tong J, Mao J (2011) NIST cloud computing reference architecture. In: Proceedings of IEEE 7th world congress on services (SERVICES'11), Washington, DC, USA, Jul. 2011, pp 594–596
5. Bokhari MU, Shallal QM, Tamandani YK (2016, March) Cloud computing service models: a comparative study. In: 3rd international conference on computing for sustainable global development (INDIACom), 16–18, March 2016, pp 890–895
6. Mahmood Z (2011, August) Cloud computing: characteristics and deployment approaches. In: 2011 IEEE 11th international conference on Computer and Information Technology (CIT), pp 121–126
7. Buyya R, Vecchiola C, Selvi ST (2013) Mastering cloud computing: foundations and applications programming. Morgan Kaufmann, USA, 2013
8. Jain N, Choudhary S (2016, March) Overview of virtualization in cloud computing. In: Symposium on colossal data analysis and networking (CDAN), pp 1–4
9. Alouane M, El Bakkali H (2016, May) Virtualization in cloud computing: no hype vs HyperWall new approach. In: 2016 International Conference on Electrical and Information Technologies (ICEIT), pp 49–54
10. Rimal BP, Choi E, Lumb I (2009, August) A taxonomy and survey of cloud computing systems. In: Fifth international joint conference on INC, IMS and IDC, 2009. NCM'09, pp 44–51
11. Afzal S, Kavitha G (2018, December) Optimization of task migration cost in infrastructure cloud computing using IMDLB algorithm. In: 2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET), pp 1–6
12. Achar R, Thilagam PS, Soans N, Vikyath PV, Rao S, Vijeth AM (2013, December) Load balancing in cloud based on live migration of virtual machines. In: 2013 annual IEEE India Conference (INDICON), pp 1–5
13. Magalhães D, Calheiros RN, Buyya R, Gomes DG (2015) Workload modeling for resource usage analysis and simulation in cloud computing. Comp Elect Eng 47:69–81

14. Dam S, Mandal G, Dasgupta K, Dutta P (2015, February) Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing. In: Proceedings of the 2015 third international conference on computer, communication, control and information technology (C3IT), pp 1–7
15. Dave A, Patel B, Bhatt G (2016, October) Load balancing in cloud computing using optimization techniques: a study. In: International Conference on Communication and Electronics Systems (ICCES), pp 1–6
16. Gupta H, Sahu K (2014) Honey bee behavior based load balancing of tasks in cloud computing. Int J Sci Res 3(6)
17. Mishra SK, Puthal D, Sahoo B, Jena SK, Obaidat MS (2017) An adaptive task allocation technique for green cloud computing. J Supercomp 405:1–16
18. Ibrahim AH, Faheem HEDM, Mahdy YB, Hedar AR (2016) Resource allocation algorithm for GPUs in a private cloud. Int J Cloud Comp 5(1–2):45–56
19. Jebalia M, Ben Letafa A, Hamdi M, Tabbane S (2015) An overview on coalitional game-theoretic approaches for resource allocation in cloud computing architectures. Int J Cloud Comp 4(1):63–77
20. Noshy M, Ibrahim A, Ali HA (2018) Optimization of live virtual machine migration in cloud computing: a survey and future directions. J Netw Comput Appl:1–10
21. Gkatzikis L, Koutsopoulos I (2013) Migrate or not? Exploiting dynamic task migration in mobile cloud computing systems. IEEE Wirel Commun 20(3):24–32
22. Jamshidi P, Ahmad A, Pahl C (2013) Cloud migration research: a systematic review. IEEE Trans Cloud Comp 1(2):142–157
23. Raviteja S, Atmakuri R, Vengaiah C (2017) A review on cloud computing migration and issues
24. Shamsinezhad E, Shahbahrami A, Hedayati A, Zadeh AK, Banirostam H (2013) Presentation methods for task migration in cloud computing by combination of Yu router and post-copy. Int J Comp Sci Iss 10(4):98
25. Ghomi EJ, Rahmani AM, Qader NN (2017) Load-balancing algorithms in cloud computing: a survey. J Netw Comput Appl 88:50–71
26. Milani AS, Navimipour NJ (2016) Load balancing mechanisms and techniques in the cloud environments: systematic literature review and future trends. J Netw Comput Appl 71:86–98
27. Kalra M, Singh S (2015) A review of metaheuristic scheduling techniques in cloud computing. Egypt Inform J 16(3):275–295
28. Mesbahi M, Rahmani AM (2016) Load balancing in cloud computing: a state of the art survey. Int J Mod Educ Comp Sci 8(3):64
29. Kanakala VR, Reddy VK, Karthik K (2015, March) Performance analysis of load balancing techniques in cloud computing environment. In: 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp 1–6
30. Shah JM, Kotecha K, Pandya S, Choksi DB, Joshi N (2017, May) Load balancing in cloud computing: methodological survey on different types of algorithm. In: 2017 International Conference on Trends in Electronics and Informatics (ICEI), pp 100–107
31. Neghabi AA, Navimipour NJ, Hosseinzadeh M, Rezaee A (2018) Load balancing mechanisms in the software defined networks: a systematic and comprehensive review of the literature. IEEE Access 6:14159–14178
32. Afzal S, Kavitha G A taxonomic classification of load balancing metrics: a systematic review
33. Vacca JR (2009) Computer and information security handbook. Morgan Kauffman, Burlington, MA, p 208
34. Kitchenham B (2004) Procedures for performing systematic reviews. Keele, UK, Keele University, 33(2004), 1–26
35. Soltani Z, Navimipour NJ (2016) Customer relationship management mechanisms: a systematic review of the state of the art literature and recommendations for future research. Comput Hum Behav 61:667–688
36. Pandey K, Swaroop A (2011) A comprehensive performance analysis of proactive, reactive and hybrid manets routing protocols. arXiv preprint arXiv:1112.5703
37. Polepally V, Chatrapati KS (2017) Dragonfly optimization and constraint measure-based load balancing in cloud computing. Cluster Comp:1–13
38. Xiao Z, Tong Z, Li K, Li K (2017) Learning non-cooperative game for load balancing under self-interested distributed environment. Appl Soft Comput 52:376–386
39. Adhikari M, Amgoth T (2018) Heuristic-based load-balancing algorithm for IaaS cloud. Futur Gener Comput Syst 81:156–165
40. Kumar M, Dubey K, Sharma SC (2018) Elastic and flexible deadline constraint load balancing algorithm for cloud computing. Proced Comp Sci 125:717–724
41. Borovskiy V, Wust J, Schwarz C, Koch W, Zeier A (2011) A linear programming approach for optimizing workload distribution in a cloud. Cloud Comp:127–132
42. Krishna PV (2013) Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl Soft Comput 13(5):2292–2303
43. Li K, Xu G, Zhao G, Dong Y, Wang D (2011, August). Cloud task scheduling based on load balancing ant colony optimization. In: 2011 sixth annual ChinaGrid conference, pp. 3–9
44. Singh A, Juneja D, Malhotra M (2015) Autonomous agent based load balancing algorithm in cloud computing. Proced Comp Sci 45:832–841
45. Lavanya M, Vaithiyanathan V (2015) Load prediction algorithm for dynamic resource allocation. Indian J Sci Technol 8(35)
46. Chen SL, Chen YY, Kuo SH (2017) CLB: a novel load balancing architecture and algorithm for cloud services. Comp Elect Eng 58:154–160
47. Ashouraei M, Khezr SN, Benlamri R, Navimipour NJ (2018, August) A new SLA-aware load balancing method in the cloud using an improved parallel task scheduling algorithm. In: 2018 IEEE 6th international conference on future internet of things and cloud (FiCloud), pp 71–76
48. Kumar M, Sharma SC (2017) Dynamic load balancing algorithm for balancing the workload among virtual machine in cloud computing. Proced Comp Sci 115(C):322–329
49. Rajput SS, Kushwah VS (2016, December) A genetic based improved load balanced min-min task scheduling algorithm for load balancing in cloud computing. In: 2016 8th international conference on Computational Intelligence and Communication Networks (CICN), pp 677–681
50. Tang L, Li Z, Ren P, Pan J, Lu Z, Su J, Meng Z (2017) Online and offline based load balance algorithm in cloud computing. Knowl-Based Syst 138:91–104
51. Ramezani F, Lu J, Hussain FK (2014) Task-based system load balancing in cloud computing using particle swarm optimization. Int J Parallel Prog 42(5):739–754
52. Vanitha M, Marikkannu P (2017) Effective resource utilization in cloud environment through a dynamic well-organized load balancing algorithm for virtual machines. Comp Elec Eng 57:199–208
53. Dasgupta K, Mandal B, Dutta P, Mandal JK, Dam S (2013) A genetic algorithm (ga) based load balancing strategy for cloud computing. Proced Technol 10:340–347
54. Cho KM, Tsai PW, Tsai CW, Yang CS (2015) A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing. Neural Comput & Applic 26(6):1297–1309
55. Dam S, Mandal G, Dasgupta K, Dutta P (2015, February) Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing. In: 2015 third international conference on computer, communication, control and information technology (C3IT), pp 1–7
56. Vasudevan SK, Anandaram S, Menon AJ, Aravinth A (2016) A novel improved honey bee based load balancing technique in cloud computing environment. Asian J Infor Technol 15(9):1425–1430
57. Kapur R (2015, August) A workload balanced approach for resource scheduling in cloud computing. In: 2015 eighth international conference on contemporary computing (IC3), pp 36–41
58. Panwar R, Mallick B (2015, October) Load balancing in cloud computing using dynamic load management algorithm. In: 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), pp 773–778
59. Sharma S, Luhach AK, Abdhullah SS (2016) An optimal load balancing technique for cloud computing environment using bat algorithm. Indian J Sci Technol 9(28)
60. Ajit M, Vidya G (2013, July) VM level load balancing in cloud environment. In: 2013 fourth International Conference on Computing,Communications and Networking Technologies (ICCCNT), pp 1–5
61. Mondal B, Choudhury A (2015) Simulated annealing (SA) based load balancing strategy for cloud computing. Int J Comp Sci Info Technol 6(4):3307–3312
62. Pasha N, Agarwal A, Rastogi R (2014) Round robin approach for VM load balancing algorithm in cloud computing environment. Int J Adv Res Comp Sci Soft Eng 4(5):34–39
63. Gulati A, Chopra RK (2013) Dynamic round robin for load balancing in a cloud computing. IJCSMC 2(6):274–278
64. Galloway JM, Smith KL, Vrbsky SS (2011, October) Power aware load balancing for cloud computing. In: proceedings of the world congress on engineering and computer science, Vol. 1, pp 19–21

65. Garg S, Gupta DV, Dwivedi RK (2016, November) Enhanced active monitoring load balancing algorithm for virtual machines in cloud computing. In: International conference on System Modeling & Advancement in Research Trends (SMART), pp 339–344
66. Tripathi AM, Singh S (2018) PMAMA: priority-based modified active monitoring load balancing algorithm in cloud computing. J Adv Res Dynam Cont Syst:809–823
67. Singh AN, Prakash S (2018) WAMLB: weighted active monitoring load balancing in cloud computing. In: Big data analytics. Springer, Singapore, pp 677–685
68. Patel G, Mehta R, Bhoi U (2015) Enhanced load balanced min-min algorithm for static meta task scheduling in cloud computing. Proced Comp Sci 57: 545–553
69. Chen H, Wang F, Helian N, Akanmu G (2013, February) User-priority guided min-min scheduling algorithm for load balancing in cloud computing. In: 2013 national conference on parallel computing technologies (PARCOMPTECH), pp 1–8
70. Mathur S, Larji AA, Goyal A (2017, June) Static load balancing using ASA max-min algorithm. Int J Res Appl Sci Eng Technol
71. Devi DC, Uthariaraj VR (2016) Load balancing in cloud computing environment using improved weighted round robin algorithm for non-preemptive dependent tasks. Sci World J

Publisher's Note