Accepted Manuscript

Load Balancing in Cloud Computing: A big Picture

Sambit Kumar Mishra, Bibhudatta Sahoo, Priti Paramita Parida

PII: S1319-1578(17)30336-1
DOI: https://doi.org/10.1016/j.jksuci.2018.01.003
Reference: JKSUCI 395

To appear in: *Journal of King Saud University - Computer and Information Sciences*

Received Date: 15 October 2017
Revised Date: 15 December 2017
Accepted Date: 15 January 2018

Please cite this article as: Mishra, S.K., Sahoo, B., Parida, P.P., Load Balancing in Cloud Computing: A big Picture, *Journal of King Saud University - Computer and Information Sciences* (2018), doi: https://doi.org/10.1016/j.jksuci.2018.01.003

Load Balancing in Cloud Computing: A big Picture ☆

[1]Sambit Kumar Mishra, [1]Bibhudatta Sahoo, [2]Priti Paramita Parida

[1] *National Institute of Technology Rourkela, India*
[2] *Department of Mathematics, Utkal University, Bhubaneswar, India*

Abstract

Scheduling or the allocation of user requests (tasks) in the cloud environment is an NP-hard optimization problem. According to the cloud infrastructure and the user requests, the cloud system is assigned with some load (that may be underloaded or overloaded or load is balanced). Situations like underloaded and overloaded cause different system failure concerning the power consumption, execution time, machine failure, etc. Therefore, load balancing is required to overcome all mentioned problems. This load balancing of tasks (those are may be dependent or independent) on virtual machines (VMs) is a significant aspect of task scheduling in clouds. There are various types of loads in the cloud network such as memory load, Computation (CPU) load, network load, etc. Load balancing is the mechanism of detecting overloaded and underloaded nodes and then balance the load among them. Researchers proposed various load balancing approaches in cloud computing to optimize different performance parameters. We have presented a taxonomy for the load balancing algorithms in the cloud. A brief explanation of considered performance parameters in the literature and their effects is presented in this paper. To analyze the performance of heuristic-based algorithms, the simulation is carried out in CloudSim simulator and the results are presented in detail.

Keywords: Cloud Computing; Energy Consumption; Load Balancing; Makespan; Virtualization; VM; Task allocation.

☆Fully documented templates are available in the elsarticle package on CTAN.

1. Introduction

In the field of network technology, the cloud computing technology is showing phenomenal growth due to the advancement of communication technology, explosive use of Internet and solve large-scale problems. It allows both hardware, and software applications as resources over the Internet for the cloud user. The cloud computing is an Internet-based computing model that share resources (e.g., networks, servers, storage, applications, and services), software, and information to various devices of the user on demand [1]. The Cloud computing is a path to utility computing embraced by some major IT companies such as Amazon, Apple, Google, HP, IBM, Microsoft, Oracle, and others. The Cloud Computing model has three service models, namely, Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS) and four deployment models, namely, private, public, hybrid and community.

The efficient and scalable features of cloud computing can achieve by maintaining proper management of cloud resources. These cloud resources are in the virtual form which is the most important characteristics of the cloud system. The Cloud Service Provider (CSP) provides services to the users in rented basis. The role of CSP to provide the services to the user is a very complex one with the available virtual cloud resources. Therefore, researchers have been given more attention towards the balancing of the load. This load balancing has a sound impact on the system performance. The CSP design a trade-off between the financial benefits and user satisfactory through load balancing. The load balancing procedure also taking care of the Service Level Agreements (SLAs), the agreement between the CSP and the cloud users.

The load balancing in clouds may be among physical hosts or VMs. This balancing mechanism distributes the dynamic workload evenly among all the nodes (hosts or VMs). The load balancing in the cloud is also referred as load balancing as a service (LBaaS). There are two versions of load balancing algorithms:

static and dynamic. The static-based balancing algorithms are mostly fit for stable environments with homogeneous system. Dynamic-based balancing algorithms are more adaptable and effective in both homogeneous and heterogeneous environment. However, the application of static load balancing procedures has less system overhead as compared to the dynamic load balancing procedures. A large number of heuristics has been proposed in the literature to this problem. Also, some metaheuristics techniques are applied to this, and here, we explained those technologies [2].

In cloud computing environment, the allocation of different tasks to VMs is known as the load. We can define the load balancing problem in various ways as follows. (1) Task allocation- The random distribution of a finite number of tasks into different Physical Machines (PMs) which again allocated to different VMs of respective PM. The efficiency of task allocation to the cloud determines the effectiveness of the load balancing algorithm [3, 4, 5]. (2) VM/Task Migration Management- In Cloud Computing Environment, VM Migration is nothing but the movement of a VM from one PM to another PM to improving the resource utilization of the data center for which the PM is overloaded. Similarly, the migration of the current state of a task from one VM to another VM or VM of one host to VM of another host is referred to as task migration. This is the reason; the VM or task migration plays a major role in load balancing of cloud computing.

In this paper, we present a review based on the modern load balancing algorithms evolved specially to suit the cloud environments. We have presented a cloud system architecture to explain the cloud system. A taxonomy is presented and elaborated for the classification of load balancing algorithms in the cloud. Various performance parameters are explained as well as compared those among different research on load balancing in a cloud.

The remaining of this paper is arranged as follows. We present the system model for cloud computing in Section 2. In Section 3, we go over various performance metrics and discuss their effects on load balancing issues in the cloud. After that, we present some load balancing approaches in Section 4, and we have compared and shown the simulation results of few heuristic-based load balancing algorithms in Section 5. Finally, we conclude the paper and expose possible areas of improvement and our future work regarding load balancing algorithms in Section 6.

2. Cloud Computing Architecture

Especially, for the design of system model, we have to consider several factors like cost, complexity, speed, system portability, security, etc. Cloud computing architecture varies from the traditional distributed system architecture in that 1) it is highly scalable, 2) it is an abstract entity and addresses distinct levels of services to the cloud consumer, 3) economies of scale control it and 4) it delivered on dynamic demand services through virtualization. One of the system architecture of a single host in the cloud environment that is followed by many researchers is shown in Figure 1.

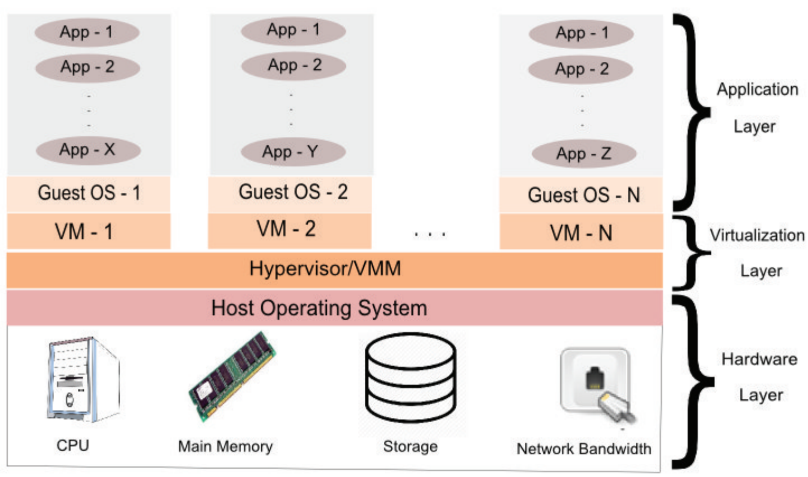

Figure 1: A Single Host Architecture in the Cloud

The hardware layer consists of hardware resources (processor, main memory, secondary storage, and network bandwidth) which are virtualized. Virtual Machine Monitor (VMM) or hypervisor (like Xen, VMWare, UML, Denali, etc.) will act as an interface between the guest operating system and VMs. This VMM support multiple operating systems to run applications on a single hardware platform concurrently. A different number of heterogeneous applications are running on each guest operating system or VM which is the basic unit to execute an application or a service request. $\{VM_1, VM_2, ..., VM_N\}$ be the set of virtual machines deployed in the cloud system as shown in Figure 1.

A cloud data center constitutes from a finite number of heterogeneous physical hosts. Each host is identified by host identification number, processing element lists, processing speed in terms of Million Instructions Per Second (MIPS), memory size, bandwidth, etc. Each host has several VMs. A VM also has same attributes like a host. The tasks arriving from the different users to the central load balancer or serial scheduler for the mapping of cloud resources. Each computing node (VM) performs execution of a single task at a time. When a request comes, it is assigned to one of the VMs by the load balancer, if sufficient resources are there to complete within the deadline. Otherwise, the task will wait if SLA permits. After the completion of the task execution, the resources used by the task on the particular VM are released and can be utilized to create new VM that can be used to serve another request.

The scheduling model in the cloud data center is shown in Figure 2. The load balancing is required due to the huge number of heterogeneous input tasks with heterogeneous resource requirement. The n number of input tasks $(T_1, T_2, ..., T_n)$ are submitted to the task queue of the cloud system. Then, the VM manager receives the input tasks from task queue and it has the complete information about the active VM, resource availability in different hosts, and the local task queue length of all the hosts. VM manager verified the resource availability of the system for a given set of input tasks. If the set of tasks can run with

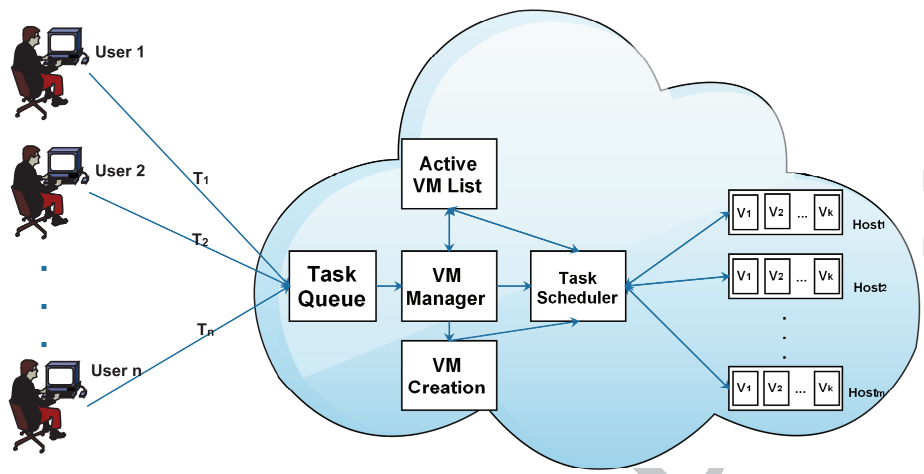

Figure 2: Scheduling Model in the Cloud Data Center

the available active VMs, then the VM manager sent those tasks to the task scheduler. Otherwise, the VM manager creates the required VMs in the host where resource availability satisfies. The task scheduler act as a load balancer where the mapping will be done among tasks and VMs based on the resource requirement of the tasks. Each host in the cloud has a finite number of active VMs.

3. Performance Matrices that effects Load Balancing

The system stability improved by balancing the load across the available virtualized resources. To have a better load balancing approach, the system requires a better scheduler. There are n input tasks and N number of virtual machines. The mapping of these n tasks to N VMs effects various system performance parameters. The finite set of user requests or tasks is $\{T_1, T_2, ..., T_n\}$. We have used Expected Time to Compute (ETC) matrix as tasks model on heterogeneous resource environment [6]. The value for ETC_{ij} is L_i/P_j, where L_i is the length of i^{th} task in terms of Million instructions (MI), and P_j is the processing speed of j_{th} VM in terms of MIPS.

Each VM has two states, i.e., active state and idle state. The idle state of VM consumes 60% of energy consumption of active state of that VM. There are two important performance parameters in cloud system : (1) makespan (MS) and (2) energy consumption (EC). The execution time of different VMs in the cloud system is different. The maximum time taken by any VM to execute all input tasks by the system is referred to as makespan of the system. The minimal makespan results in a better balancing of the load. The execution time of j^{th} VM (ET_j) is based on the decision variable X_{ij}, where

$$X_{ij} = \begin{cases} 1 \text{ if } T_i \text{ is allocated to } VM_j \\ 0 \text{ if } T_i \text{ is not allocated to } VM_j \end{cases} \quad (1)$$

And the $ET_j, 1 \leq j \leq N$ is calculated as follows.

$$ET_j = \sum_{i=1}^{n} X_{ij} \times ETC_{ij} \quad (2)$$

The makespan (MS) is the maximum of ET_j i.e.,

$$MS = Max.[ET_j]_{j=1}^{N} \quad (3)$$

Suppose, the virtual machine VM_j consumes α_j Joules/MI in active state and β_j Joules/MI in idle state. The VM_j remains in idle state for $MS - ET_j$ seconds. So, the total energy consumption (EC i.e., the sum of energy consumption in the active and idle state) is in equation (4) as follows.

$$EC = \sum_{j=1}^{N} [[ET_j \times \alpha_j + (MS - ET_j) \times \beta_j] \times P_j] \quad (4)$$

Some important performance metrics including makespan and energy consumption that effects load balancing in cloud computing are explained below. The detail approach used in different Load Balancing Algorithms in various simulation environments is explained in Table 1, Table 2, and Table 3.

Throughput (TP): Throughput indicates the number of user requests (tasks) executed per unit time by a virtual machine. Throughput value determines the system performance. High throughput implies that the system performance is good. The throughput of the system is inversely proportional to the makespan of the system.

Thrashing (TH): Thrashing will occur due to memory, or other resources have become exhausted or too limited in the system to perform operations on user requests. In the cloud environment, it occurs when the number of VM is spending their time in migration without maintaining the proper scheduling. So, the appropriate load balancing algorithm used for the maintenance of this factor.

Reliability (R): Reliability will consistently performs according to the system specifications. In case of any failure during task execution, the task is transferred to any other resources (VMs) to improve reliability of the system. The reliable system improves the stability of the system.

Accuracy (A): It determines the perfection of task execution result. Accuracy is the ability of a measurement that can match with the actual value of the task execution being measured. In the current time, IT-industries give more importance on the system accuracy according user demand. The accuracy value slightly degrades the system makespan.

Predictability (PR): It is the degree used for the prediction of task allocation, task execution, and task completion according to the available cloud resources (virtual machines). The previous behavior of the arrival of task to the system and the allocation and execution of those tasks in the cloud system provides the predictability value. The better prediction of task allocation advances the balancing of the load as well as enhances the system makespan.

Makespan (MS): It is the total time required to complete all tasks submitted to the system. Makespan of the system is the maximum time taken by the host running over the data center. In some cases where tasks have some priorities to execute, then the CSP has to compromise for the makespan of the system.The evaluation of makespan (MS) of the system is exhibited in equation (3). The optimal makespan results in proper system load balancing.

Scalability(S): It is a feature of a system or model to describes the capability of the system to perform under unexpected circumstances. It means the level of surviving for a balanced system when the number or size of task or workload is increased. In a scalable cloud system, the rescaling of resources will be done periodically.

Fault Tolerance (FT): The fault-tolerant method is one of the capabilities of the system to perform uninterrupted service even if one or more system elements failing. It also resolves obstacles related to logical errors. The level of fault-tolerance can be measured from the number of failure points (i.e., single point failure or multipoint failure). To overcome some failures of the cloud system, the service providers require additional resources or virtual machines. However, this process leads to some additional costs, but the user can get a fault-free system.

Associated Overhead (AO): It is the overhead formed by the execution of the algorithms. The balancing technique for the load to the system results in some overhead cost. If the load of the system is balanced properly, then minimum overhead occurs [7].

Migration time (MT): The actual time required to migrate a task or VM from one resource to another. The migration of task may be from one VM to another VM within a single host or different host. Similarly, the migration of VM will be from one host to another host within a data center or in different data centers. When a task required resources through multiple virtual machines or the execution of the task is interrupted, then the task is migrated. In the same way, if a VM is crash during execution, then the VM is migrated to another host. The higher number of migration of VMs results in more migration time which degrades the makespan and

Table 1: Detail approach used in different Load Balancing Algorithms

Algorithm	Approaches	Simulator	Environment
[10]	Introduced a scheduling strategy based on Load Balancing Ant Colony Optimization (LBACO) algorithm where they balance the system load while trying to reduce the makespan.	CloudSim	Homogeneous
[11]	GA and Gravitational Emulation Local Search (GELS) have used for load balancing of VMs.	Cloud-Analyst	Homogeneous
[12]	Presented Autonomous Agent-Based Load Balancing Algorithm (A2LB) for dynamic load balancing in the cloud.	Java	Homogeneous
[13]	Priority scheduling and convex optimization theory have used to avoid cluster load balancing problem.	NA	Heterogeneous
[14]	Used for homogeneous resource allocation for speed network.	Gridsim	Homogeneous
[15]	Proposed load balancing techniques for Distributed virtual environments (DVEs) based on heat diffusion where they examine two special factors, the convergence threshold and the load balancing interval.	C++	Heterogeneous
[16]	The f-restricted First Fit Algorithm vectors used for resource allocation.	NA	Homogeneous
[17]	Used Honeybee concept for assigning the available resources to the network to reduce makespan.	CloudSim & Work-flow	Heterogeneous
[18]	Enhanced Exponentially Weighted Moving Average (EEWMA) method is applied to predict the load, and also suitable to predict the future resource demand considering scalability issue.	Real time Implementation	Heterogeneous
[19]	An Improve PSO is used for enhanced efficiency and speed of task.	MATLAB	Heterogeneous
[20]	Proposed a load balancing approach for cloud computing, Load Balanced Resource Scheduling Algorithm (LBRS) by considering two categories of resources: reservation resources and On-Demand resources.	PHP and MySQL	Both

Table 2: Detail approach used in different Load Balancing Algorithms

Algorithm	Approaches	Simulator	Environment
[21]	Proposed a heuristic algorithm that uses Tabu Search technique and task grouping with prioritization .	CloudSim	Heterogeneous
[22]	Presented a Simple Scheduling Algorithm with Load Balancing (SSALB), which reduces the makespan and maximum resource utilization.	Real time Implementation	Homogeneous
[23]	Proposed a fuzzy logic-based load balancing technique that performs without any future knowledge to reduce energy and cost.	CloudSim, Google cluster	Heterogeneous
[24]	Proposed a modified Bin-Packing model for the optimal resource (VM) placement in the cloud as a maximum flow problem to serve multiple user demands.	C++	Homogeneous
[25]	Proposed a load balancing algorithm to allocate the dynamic load uniformly at the servers by analyzing the current status of all the possible VMs and also calculates the response time of their algorithm.	Cloud-Analyst	Homogeneous
[26]	Proposed a fully distributed load rebalancing technique and also that technique minimizes the network traffic (loads of nodes) by maximizing the network bandwidth.	Hadoop System	Both
[27]	Used Bat algorithm to search the optimal host as well as VM for an incoming task. The task is submitted to the searched host otherwise distributed among multiple servers.	MATLAB	Heterogeneous
[28]	Proposed Synchronized Throttled Load Balancing (STVMLB) algorithm used to reduce the overload or underload on VMs and also to maximize the VM utilization in the cloud computing.	Cloud-Analyst	Homogeneous

Table 3: Detail approach used in different Load Balancing Algorithms

Algorithm	Approaches	Simulator	Environment
[29]	Proposed a self-adaptive Randomized Optimization Approach to balance the load of servers in the cloud system.	Lighttpd2 web server, Xen hypervisor	Homogeneous
[30]	Proposed a VM load balancer algorithm which guarantees uniform allocation of tasks to VMs even during peak times when the number of tasks received is very high.	Cloud-Analyst	Homogeneous
[31]	Proposed a Weighted Signature based load balancing (WSLB) algorithm to reduce response time. WSLB obtain the load assignment factor for each host and map the virtual machines as stated in factor value.	Cloud-Analyst	Homogeneous
[32]	Proposed EcoPower, an online algorithm to achieve eco-aware power management and load scheduling collectively for cloud data centers, and employ the Lyapunov optimization theory to compose their algorithm.	Real time Implementation	Heterogeneous
[33]	Propose a load balancing algorithm based on the process of evaluating the task completion time to enhance the system performance concerning processing time and response time.	CloudSim	Heterogeneous
[34]	A soft computing strategy based algorithm SA is used to solve the load balancing problem for dynamic workload across various resources.	Cloud-Analyst	Heterogeneous

Table 4: Load balancing Algorithms and their corresponding performance metrics

Algorithm	TP	TH	R	A	PR	MS	S	FT	AO	MT	RT	AC	EC
[10]	×	√	×	√	√	√	×	×	×	√	√	√	×
[11]	√	√	×	×	×	√	√	√	×	×	×	√	×
[12]	×	×	√	×	×	×	√	×	×	×	√	√	×
[13]	×	×	×	×	×	×	√	×	×	×	√	√	√
[14]	√	√	×	√	×	×	√	×	√	√	√	√	×
[15]	×	×	×	×	√	×	×	√	√	√	√	√	×
[16]	×	×	×	√	×	√	×	×	×	√	×	√	√
[17]	×	×	√	√	√	√	√	√	√	√	×	√	×
[18]	√	×	√	√	×	×	√	√	√	√	√	√	√
[19]	√	×	×	√	×	√	×	×	×	×	×	√	×
[20]	√	×	√	×	×	×	×	×	×	×	√	√	√
[21]	√	×	√	×	×	√	√	×	×	×	√	√	×
[22]	×	√	×	×	×	√	×	×	√	×	×	×	×
[23]	×	×	×	×	×	×	×	×	×	√	×	√	√
[24]	×	×	×	×	×	×	√	×	×	×	√	√	√
[25]	√	×	√	√	√	×	√	×	×	×	√	√	×
[26]	×	×	×	×	×	×	×	×	√	×	×	√	×
[27]	×	×	×	×	×	×	×	×	×	√	√	×	×
[28]	√	×	×	×	√	×	√	×	√	×	√	×	×
[29]	×	×	×	×	√	×	√	×	×	√	√	√	√
[30]	×	×	√	×	√	×	×	×	×	√	√	√	×
[31]	×	×	×	×	×	×	×	×	×	×	√	√	×
[32]	√	×	×	×	×	×	×	×	√	×	×	√	√
[33]	√	×	×	×	×	×	×	×	×	√	×	×	√
[34]	×	×	×	×	×	×	×	×	×	×	√	√	×

load balancing of the system.

Response time (RT): It is the time required by the system to respond a task. In other words, it is the sum of transmission time, waiting time, and service time. Thus, the system performance is inversely proportional to the response time. The optimal response time results in a better makespan value.

Associated Cost (AC): This cost depends on the percentage of resource utilization. For example, the services offered by EC2 can reduce the entire cost up to 49 % while the resource is fully utilized ([8]). The cloud user tries to depreciate the cost of resource provisioning by degrading the on-demand resource cost and over-subscribed resource cost of over-provisioning and under-provisioning [9].

Energy Consumption (EC): The energy consumption of a cloud system is the amount of energy absorbed by all ICT devices connected in the system [35]. Three kinds of devices to calculate the energy consumption are personal terminals (desktop, laptop, handsets, etc.), networking nodes (routers, switches, hubs, etc.), local servers (application servers). There are four different solutions to conserve energy, and those are the use of energy-efficient hardware, application of energy-aware scheduling technique, power-minimization in the server cluster, and power-minimization in wired and wireless networks [36]. The estimation of energy consumption of the system is presented in equation (4) based two state virtual machines.

The load balancing algorithms and their corresponding consideration of different performance metrics is presented in Table 4. These metrics indicate the performance of different load balancing algorithms.

4. Classification of load balancing Algorithm

The task allocation algorithms in the cloud are classified based upon the current state of VM. In allocation policy where the current load information of VMs are available before the allocation is said to be a dynamic strategy. Whereas the static strategy acts on VMs without any load information. Load balancing attends in fair allocation of resources to achieve a high user satisfaction and improve the stability of the system. We have proposed a taxonomy for the load balancing algorithms in the cloud environment as shown in Figure 3. Resource management plays a major role in the load balancing of cloud resources [37].

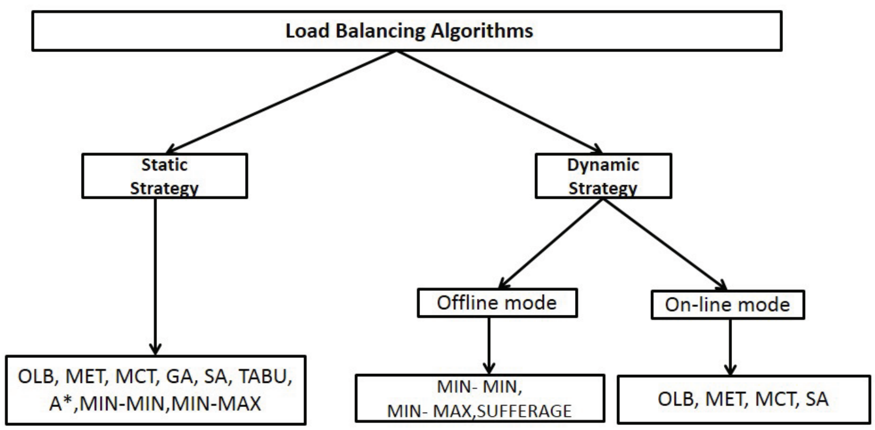

Figure 3: Classification of load balancing Algorithm

4.1. Load Balancing Algorithms

Normally, the load balancing in cloud computing with a multi-objective system is a well known NP-complete problem [10, 11]. The objectives may be energy saving, makespan minimization, throughput maximization, etc. Researchers presented different heuristic techniques (or sub-optimal algorithms) to obtain a sub-optimal solution for load balancing in the cloud environment. To balance the load of the system, load balancers are required. The detail explanation of various load balancers are presented in Table 5. We have studied two types of heuristic strategies for the load balancing in cloud computing i.e. static and dynamic.

Static strategies: Typically, the static load in cloud computing strategies are coming under two assumptions. The first is the initial task arrival and the second is the availability of physical machines at the beginning. The resource update will be carried out after each task is scheduled. Some presented heuristics in static strategy are OLB, MET, MCT, GA, Switching Algorithm, TABU, A* algorithm, Min-Min, Min-Max.

Dynamic strategies: It is an important strategy in cloud computing environment because the load distributed among physical machines during the run-time. Here, the arrival time of tasks is unusual, and the creation of virtual machines is also according to the type of input tasks.

These heuristics based dynamic algorithms applied for load balancing can be classified into two categories: Off-line mode (Batch mode) and On-line mode. In batch mode heuristics, the task is allocated only at some predefined moments. It is used to determine the actual execution time of a larger number of tasks. Some presented heuristics for batch modes are Max-min, Min-min, Sufferage algorithm. In on-line mode (or immediate mode), a user request (task) is mapped onto a computing node as soon as it enters at the scheduler. Here, each task is scheduled only once, the scheduling result remains unchanged. Some presented heuristics for online modes are OLB, MET, MCT, SA.

OLB (Opportunistic load balancing): OLB heuristic technique is used through both static and dynamic (Online mode) strategy in a cloud environment. This heuristic always allocates tasks to virtual machines arbitrarily and

Table 5: Detail explanation of different Load Balancers

Load Balancer	Description
Hardware load Balancer (HLD)	HLD is a physical unit that manages the individual server in a network and used to spread web traffic over multiple network servers. LBaaS is an alternative to HLD. It offers global server load balancing and is fit for the heterogeneous environment.
Network Load Balancer (NLB)	NLB serves at the network layer or Layer 4 of OSI model. It is perfect for load balancing of TCP traffic [53]. NLB provides a static IP for every subnet which can be utilized by the applications as the front-end IP of the balancer. It is used for distributes network traffic in various VMs within a cluster to avoid overloading.
Application Load Balancer (ALB)	ALB serves at the layer 7 of OSI model. It is absolute for high-level load balancing of HTTP and HTTPS traffic [53]. ALB interprets and improves the protection of the application, by assuring the advanced SSL/TLS ciphers and protocols are employed every time.
Classic Load Balancer (CLB)	CLB affords fundamental load balancing over multiple Elastic Cloud Compute (EC2) instances [53]. It works at both the request level and connection level. CLB is designed for the EC2-Classic network applications.
Elastic Load Balancer (ELB)	It is also known as AWS load balancer. It distributes incoming task over multiple Amazon EC2 instances. It offers three kinds of load balancers: Application Load Balancer (ALB), Network Load Balancer (NLB), and Classic Load Balancer (CLB) [53].
HAProxy Load Balancer (HAPLB)	Its configuration has two interfaces: one towards users and another towards the server LAN. The HAPLB also serves in Layer 4 and Layer 7 of OSI model. It is mainly used in reverse proxy or ALOHA load balancer. The ALOHA Load-Balancer provides scalable and reliable infrastructures. The ALOHA Load-Balancer developed several open source load balancing software utilizing HAProxy.

then checks for the next available machine. In online mode, each task assigned to the host based on various parameters like execution time of the task on that machine. The task execution will be done in VM-level. According to [38] and [39], the OLB scheduling algorithm is used to allocate the task and divides a task into subtasks in a three level cloud computing network (i.e., Request manager, Service manager, Service node) for assigning and solving the workload in the least time. It does not take additional calculations for the allocation and load balancing of tasks; it considers overall expected completion time to execute a task. They have measured the makespan of the system through the algorithm. The merit of OLB is to keep all hosts busy as much as possible which shows better efficiency and maintain proper balancing of the load for the system. OLB is not suitable for cloud environment due to poor make-span when multiple objectives are considered simultaneously.

MET (Minimum Execution Time): MET is also known as LBA (Limited Best Assignment) [40] or UDA (User Directed Assignment) [41]. This heuristic technique used in both static and dynamic (Online mode) strategy. This algorithm was presented in [6] to map each task to the virtual machine. The scheduler assigns each task according to lowest execution time as in Expected Time to Compute (ETC) matrix to the VM so that the system performs all tasks with the execution time. Maheswaran *et al.* [6] have tried to enhance makespan of the system through the allocation of tasks with some balancing of cloud resources. The main fault of this technique is that it does not consider machine ready time and shows several variances in load across the machine.

MCT (Minimum Compilation Time): MCT heuristic technique is used in both static and dynamic (Online mode) load balancing strategy. Kim *et al.* [42] have used MCT technique where they considered both ready-to-execute time and the expected execution time of the tasks for balancing purpose. In that, they allocate the task to the core that has least completion time. The MCT will perform after allocation of task to a machine for the selection of appropriate core.

Min-Min: The basic Min-Min procedure in cloud environment selects the task with least size and chose a cloud resource (VM) that has the minimum capacity. After allocation of a task to a VM, that task is removed from the queue and proceed forward for the distribution of rest all unallocated tasks. The Min-Min algorithm is only suitable for small-scale Distributed systems [43]. Chen *et al.* [44] have introduced an improved Min-Min algorithm to balance the load of the system as well as to optimize the makespan and enhance the resource utilization. The Load Balance Improved Min-Min (LBIMM) algorithm proposed by them will first split all the tasks into two groups A and B. A is for the higher priority tasks and B is for the lower priority tasks. Then, the algorithm schedules all the tasks A first and then moves to the allocation of tasks in B. Finally, the load balancing function is operated to optimize the particular load of each machine to generate the final schedule.

Min-Max: Max-Min is similar to the Min-Min heuristic algorithm. The basic Max-Min procedure in cloud environment selects the task with larger size and chose a cloud resource (VM) that has the minimum processing capacity. After allocation of a task to a VM, that task is removed from the queue and proceed forward for the distribution of rest all unallocated tasks. The Max-Min algorithm is also suitable for only small-scale Distributed systems [43]. To accomplish the balancing of load, Li *et al.* [45] have intended an augmented Max-Min algorithm that keeps a task status table to measure the real-time load of VMs as well as the expected completion time of tasks. The Elastic Cloud Max-Min (ECMM) algorithm proposed by [45] is better than the round robin technique for the consideration of average task pending time. In that, the tasks arrived at the system in batch process.

Genetic Algorithm (GA): GA is based on the population and individual chromosome (Possible allocation) which may have some fitness values (like energy consumption, makespan, throughput, etc.) for optimizing them. The basic GA algorithm performs Selection (e.g. Roulette wheel selection, Tournament selection, etc.), Crossover (e.g. Matrix Crossover, Single point crossover, etc.), Mutation (e.g. bit-wise mutation, boundary mutation, etc.) in each iteration. Most of the researchers considered the chromosome (is in vector form) size as the total number of tasks arrived at the system. Dasgupta *et al.* [46] have proposed a GA-based load balancing approach that minimizes the makespan. They encoded the population with binary strings, and the chromosomes experience a random single point crossover, and considered 0.05 as mutation probability.

Simulated annealing (SA): SA is a method for resolving unconstrained and bound-constrained optimization problems. At each iteration of the algorithm, a new point is generated based on a probability distribution. The algorithm avoids being confined in local minima and is able to search globally for good solutions [47]. Moschakis *et al.* [48] have used SA-based method for the consolidation of various jobs to the available resources.

Tabu Search (TS): TS is a meta-heuristic based local heuristic to explore the solution space ahead local optimality. This method uses adaptive memory that performs a more elastic search behavior [49]. Tsai *et al.* [2] have presented a parallel variant of TS which applied the master-slave model. Tsai *et al.* [50] have presented an efficient TS heuristic for placing the cloud data centers in different locations. Their primary objectives are to enhancing the network performance, reducing the CO_2 emissions, and optimizing the resource utilization cost. The effectiveness of the TS is examined for networks with up to 500 nodes and 1,000 data center locations.

A-star Search: A-star search algorithm is extensively applied as a graphic searching algorithm. This heuristic algorithm combines the benefits of both depth-first search and breadth-first search algorithm. It supports two lists, the first list act as a priority queue of the tasks and the second list has the processing capacity of all VMs. AlShawi *et al.* [51] have presented a procedure to enlarge network lifetime applying a combination of a fuzzy method and an A-star algorithm.

Switching Algorithm: This algorithm is used in the cloud environment for the migration of tasks or VMs. Using this method, we can achieve the fault-tolerant property. Shao *et al.* [52] have proposed a switching algorithm for the switching of tasks to balance the load.

5. Simulation Results

Besides the classification of load balancing algorithms as shown in Figure 3, each algorithm or technique must be of heuristic or metaheuristic type. Here, some examples of heuristic algorithms are OLB, MCT, MET, Min-Min, Min-Max, etc., and some examples of metaheuristic algorithms are GA, SA, TS, etc. In this paper, we have shown some simulation results to analyze the performance of few heuristic-based scheduling algorithms. We have analyzed the load balancing algorithms (MCT, MET, Min-Min, Max-Min, and Min-Max) through simulation with generated datasets. The experiments were performed using CloudSim-3.0.3 simulator [54]. The version of the system is Intel Core i7 4th Generation processor, 3.4 GHz CPU and 8GB RAM running on Microsoft Windows 8 platform. The arrival rate of the task follows the Pareto distribution. Here, to analyze the algorithms, we have considered makespan and energy consumption of the system as performance metrics. We have conducted two sets of simulation scenarios as follows.

Scenario-1: For this scenario, the total number of tasks is 500 which is fixed. The number of VMs varies from 20 to 200 in intervals of 20. A comparative report is shown in Figure 4 and Figure 5. The bar chart in Figure 4 and Figure 5 shows that the makespan and energy consumption minimum for the MCT load balancing algorithm among the compared five algorithms.

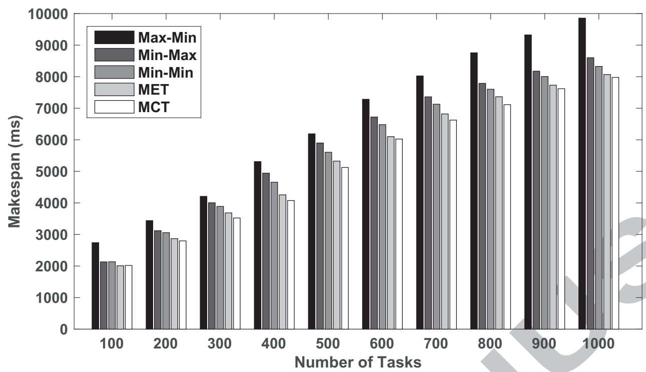

Figure 4: Makespan comparison of Max-Min, Min-Max, Min-Min, MET, and MCT load balancing algorithms for scenario-1

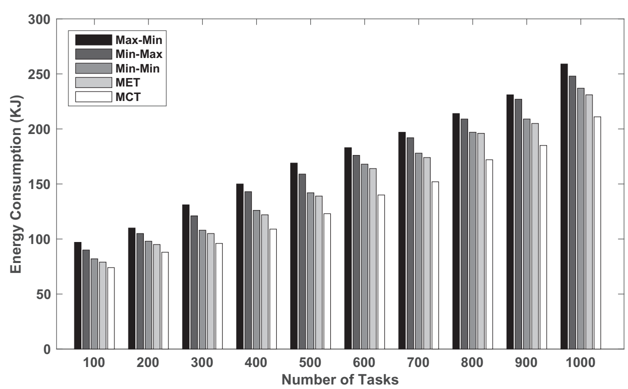

Figure 5: Energy Consumption comparison of Max-Min, Min-Max, Min-Min, MET, and MCT load balancing algorithms for scenario-1

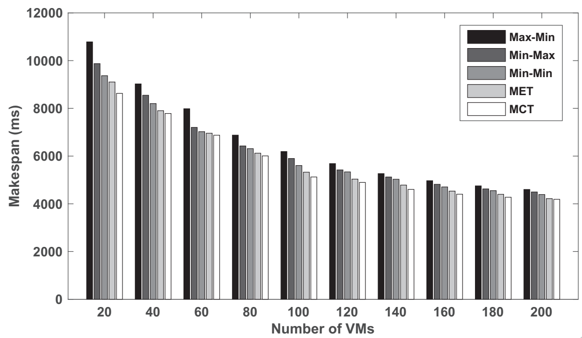

Figure 6: Makespan comparison of Max-Min, Min-Max, Min-Min, MET, and MCT load balancing algorithms for scenario-2

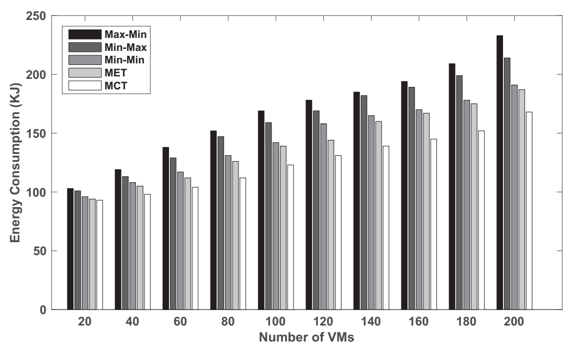

Figure 7: Energy Consumption comparison of Max-Min, Min-Max, Min-Min, MET, and MCT load balancing algorithms for scenario-2

Scenario-2: For this scenario, the total number of VMs is 100 which is fixed. The number of input tasks varies from 100 to 1000 in intervals of 100. A comparative report is shown in Figure 6 and Figure 7. The bar chart in Figure 6 and Figure 7 shows that the makespan and energy consumption minimum for the MCT load balancing algorithm among the compared five algorithms. Here, in both the scenarios, the Max-Min load balancing algorithm not performed better as compared to the MCT, MET, Min-Min, and Min-Max algorithms.

6. Conclusion

In this study, we have described various load balancing techniques in different (i.e., homogeneous, heterogeneous) cloud computing environments. A system architecture, with distinct models for the host, VM is described. We have explained various performance parameters listed in the above tables those evaluate the system performance. The calculation of makespan and energy consumption of the system is explained in details. We have proposed a taxonomy for the load balancing algorithm in the cloud environment. To analyze the performance of heuristic-based algorithms, the simulation is carried out in CloudSim simulator and the results are presented in detail. For further researches, understanding of these approaches is essential.

Future work includes evaluating the proposed algorithms in a real-world cloud deployment, and also implementation of all discussed techniques and make a comparison among all.

References

[1] Bohn, R. B., Messina, J., Liu, F., Tong J. and Mao J. (2011) 'NIST Cloud Computing Reference Architecture', *IEEE World Congress on Services, Washington, DC*, pp. 594–596.

[2] Tsai, C. W., and Rodrigues, J. J. (2014) 'Metaheuristic scheduling for cloud: A survey', *IEEE Systems Journal*, 8(1), pp. 279-291.

[3] Mishra, S. K., Puthal, D., Sahoo, B., Jena, S. K., and Obaidat, M. S. (2017) 'An adaptive task allocation technique for green cloud computing', *The Journal of Supercomputing*, pp. 1-16.

[4] Ibrahim, A. H., Faheem, H. E. D. M., Mahdy, Y. B., and Hedar, A. R. (2016) 'Resource allocation algorithm for GPUs in a private cloud', *International Journal of Cloud Computing*, 5(1-2), pp. 45-56.

ACCEPTED MANUSCRIPT

[5] Jebalia, M., Ben Letafa, A., Hamdi, M., and Tabbane, S. (2015) 'An overview on coalitional game-theoretic approaches for resource allocation in cloud computing architectures', *International Journal of Cloud Computing*, 22, 4(1), pp. 63-77.

[6] Maheswaran, M., Ali, S., Siegal, H. J., Hensgen D. and Freund, R. F. (1999) 'Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems', *Eighth Heterogeneous Computing Workshop, 1999. (HCW '99) Proceedings, San Juan*, pp. 30–44.

[7] Singh, A., Juneja, D., and Malhotra, M. (2015) 'A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing', *Journal of King Saud University-Computer and Information Sciences*.

[8] Amazon EC2 Reserved Instances, http://aws.amazon.com/ec2/reserved-instances, 2012.

[9] Chaisiri, S., Lee, B. S., and Niyato, D. (2012) 'Optimization of resource provisioning cost in cloud computing', *IEEE Transactions on Services Computing*, 5(2), pp. 164–177.

[10] Li, K., Xu, G., Zhao, G., Dong, Y. and Wang, D. (2011) 'Cloud Task Scheduling Based on Load Balancing Ant Colony Optimization', *Sixth Annual China grid Conference, Liaoning*, pp. 3–9.

[11] Dam, S., Mandal, G., Dasgupta, K., and Dutta, P. (2015, February) 'Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing', *Third IEEE International Conference on Computer, Communication, Control and Information Technology (C3IT)*, pp. 1–7.

[12] Singh, A., Juneja, D., and Malhotra, M. (2015) 'Autonomous agent based load balancing algorithm in cloud computing', *Procedia Computer Science*, 45, pp. 832–841.

[13] Wang, R., and Rao, B. (2015) 'Research on the Cloud Computing Load Balance Degree of Priority Scheduling Algorithm based on Convex Optimization Theory', pp. 156–160.

[14] Patni, J. C., Aswal, M. S., Agarwal, A., and Rastogi, P. (2015) 'A Dynamic and Optimal Approach of Load Balancing in Heterogeneous Grid Computing Environment', *In Emerging ICT for Bridging the Future-Proceedings of the 49th Annual Convention of the Computer Society of India CSI, Springer*, Vol. 2, pp. 447–455.

[15] Deng, Y., and Lau, R. W. (2014) 'Dynamic load balancing in distributed virtual environments using heat diffusion', *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 10(2), pp. 16.

[16] Azar, Y., Cohen, I. R., Fiat, A., and Roytman, A. (2016, January) 'Packing small vectors', *In Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1511–1525.

[17] Vasudevan, S. K., Anandaram, S., Menon, A. J., and Aravinth, A. (2016), 'A Novel Improved Honey Bee Based Load Balancing Technique in Cloud Computing Environment', *Asian Journal of Information Technology*, 15(9), pp. 1425–1430.

[18] Lavanya, M., and Vaithiyanathan, V. (2015) 'Load Prediction Algorithm for Dynamic Resource Allocation', *Indian Journal of Science and Technology*, 8(35), pp. 1–4.

[19] Zhu, Y., Zhao, D., Wang, W., and He, H. (2016, January) 'A Novel Load Balancing Algorithm Based on Improved Particle Swarm Optimization in Cloud Computing Environment', *In International Conference on Human Centered Computing, Springer*, pp. 634–645.

[20] Kapur, R. (2015, August) 'A workload balanced approach for resource

scheduling in cloud computing', *Eighth IEEE International Conference on Contemporary Computing (IC3)*, pp. 36–41.

[21] Alam, M. I., Pandey, M., and Rautaray, S. S. (2014) 'A Proposal of Resource Allocation Management for Cloud Computing', *International Journal of Cloud Computing and Services Science*, 3(2), pp. 79–86.

[22] Alharbi, F., and Rabigh, K. S. A. (2012) 'Simple scheduling algorithm with load balancing for grid computing', *Asian Transactions on Computers*, 2(2), pp. 8–15.

[23] Toosi, A. N., and Buyya, R. (2015, December) 'A Fuzzy Logic-based Controller for Cost and Energy Efficient Load Balancing in Geo-Distributed Data Centers', *In 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, pp. 186–194.

[24] Hadji, M., and Zeghlache, D. (2012, June) 'Minimum cost maximum flow algorithm for dynamic resource allocation in clouds', *IEEE 5th International Conference on Cloud Computing (CLOUD)*, pp. 876–882.

[25] Panwar, R., and Mallick, B. (2015, October) 'Load balancing in cloud computing using dynamic load management algorithm', *IEEE International Conference on Green Computing and Internet of Things (ICGCIoT)*, pp. 773–778.

[26] Hsiao, H. C., Chung, H. Y., Shen, H., and Chao, Y. C. (2013) 'Load rebalancing for distributed file systems in clouds', *IEEE transactions on parallel and distributed systems*, 24(5), pp. 951–962.

[27] Sharma, S., Luhach, A. K., and Abdhullah, S. S. (2016) 'An Optimal Load Balancing Technique for Cloud Computing Environment using Bat Algorithm', *Indian Journal of Science and Technology*, 9(28), pp. 1–4.

[28] Garg, S., Dwivedi, R. K., and Chauhan, H. (2015, September) 'Efficient utilization of virtual machines in cloud computing using Synchronized Throt-

tled Load Balancing', *1st IEEE International Conference on Next Generation Computing Technologies (NGCT)*, pp. 77–80.

[29] Papadopoulos, A. V., Klein, C., Maggio, M., Drango, J., Dellkrantz, M., Hernndez-Rodriguez, F., ... and rzn, K. E. (2016) 'Control-based load-balancing techniques: Analysis and performance evaluation via a randomized optimization approach', *Control Engineering Practice*, 52, pp. 24–34.

[30] Kulkarni, A. K., and Annappa, B. (2015, February) 'Load balancing strategy for optimal peak hour performance in cloud datacenters', *In IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, pp. 1–5.

[31] Ajit, M., and Vidya, G. (2013, July) 'VM level load balancing in cloud environment', *In IEEE Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pp. 1–5.

[32] Deng, X., Wu, D., Shen, J., and He, J. (2016) 'Eco-aware online power management and load scheduling for green cloud datacenters', *IEEE Systems Journal*, 10(1), pp. 78–87.

[33] Chien, N. K., Son, N. H., and Loc, H. D. (2016, January) 'Load balancing algorithm based on estimating finish time of services in cloud computing', *In 2016 18th IEEE International Conference on Advanced Communication Technology (ICACT)*, pp. 228–233.

[34] Mondal, B., and Choudhury, A. (2015) 'Simulated Annealing (SA) based Load Balancing Strategy for Cloud Computing', *(IJCSIT) International Journal of Computer Science and Information Technologies*, 6(4), pp. 3307–3312.

[35] Moganarangan, N., Babukarthik, R. G., Bhuvaneswari, S., Basha, M. S., and Dhavachelvan, P. (2016) 'A novel algorithm for reducing energy-consumption in cloud computing environment: Web service computing ap-

proach', *Journal of King Saud University-Computer and Information Sciences*, 28(1), pp. 55-67.

[36] Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M. Q., and Pentikousis, K. (2010) 'Energy-efficient cloud computing', *The computer journal*, 53(7), pp. 1045–1051.

[37] Al-Ayyoub, M., Daraghmeh, M., Jararweh, Y., and Althebyan, Q. (2016) 'Towards improving resource management in cloud systems using a multi-agent framework', *International Journal of Cloud Computing*, 5(1-2), pp. 112-133.

[38] Wang, S.C., Yan, K. Q., Liao, W. P. and Wang, S. S. (2010) 'Towards a Load Balancing in a three-level cloud computing network', *3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), Chengdu*, pp. 108–113.

[39] Rewehel, E. M., Mostafa, M. S. M., and Ragaie, M. O. (2014, October) 'New Subtask Load Balancing Algorithm Based on OLB and LBMM Scheduling Algorithms in Cloud', *In Proceedings of the 2014 International Conference on Computer Network and Information Science, IEEE Computer Society*, pp. 9–14.

[40] Kanakala, V., RaviTeja, V., Reddy, K. and Karthik, K. (2015) 'Performance analysis of load balancing techniques in cloud computing environment', *IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1–6.

[41] Armstrong, R., Hensgen, D. and Kidd, T. (1998) 'The relative performance of various mapping algorithms is independent of sizable variances in run-time predications', *7th IEEE Heterogeneous Computing Workshop (HCW98)*, Mar. 1998, pp. 79-87.

[42] Kim, S. I., Kim, H. T., Kang, G. S., and Kim, J. K. (2013, June) 'Using dvfs and task scheduling algorithms for a hard real-time heterogeneous

multicore processor environment', *In Proceedings of the 2013 workshop on Energy efficient high performance parallel and distributed computing, ACM*, pp. 23-30.

[43] Kokilavani, T., and Amalarethinam, D. D. G. (2011) 'Load balanced min-min algorithm for static meta-task scheduling in grid computing', *International Journal of Computer Applications*, 20(2), pp. 43-49.

[44] Chen, H., Wang, F., Helian, N., and Akanmu, G. (2013, February) 'User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing', *In IEEE National Conference on Parallel Computing Technologies (PARCOMPTECH)*, pp. 1-8.

[45] Li, X., Mao, Y., Xiao, X., and Zhuang, Y. (2014, June) 'An improved max-min task-scheduling algorithm for elastic cloud', *In IEEE International Symposium on Computer, Consumer and Control (IS3C)*, pp. 340-343.

[46] Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., and Dam, S. (2013) 'A genetic algorithm (ga) based load balancing strategy for cloud computing', *Procedia Technology*, 10, pp. 340-347.

[47] Hwang, C. R. (1988) 'Simulated annealing: theory and applications', *Acta Applicandae Mathematicae*, 12(1), pp. 108-111.

[48] Moschakis, I. A., and Karatza, H. D. (2015) 'Towards scheduling for Internet-of-Things applications on clouds: a simulated annealing approach', *Concurrency and Computation: Practice and Experience*, 27(8), pp. 1886-1899.

[49] Glover, F., and Laguna, M. (2013) 'Tabu Search', *Springer New York*, pp. 3261-3362.

[50] Larumbe, F., and Sanso, B. (2013) 'A tabu search algorithm for the location of data centers and software components in green cloud computing networks', *IEEE Transactions on Cloud Computing*, 1(1), pp. 22-35.

[51] AlShawi, I. S., Yan, L., Pan, W., and Luo, B. (2012) 'Lifetime enhancement in wireless sensor networks using fuzzy approach and A-star algorithm', *IEEE Sensors journal*, 12(10), pp. 3010-3018.

[52] Shao, S., Guo, S., Qiu, X., and Meng, L. (2014, August) 'A random switching traffic scheduling algorithm in wireless smart grid communication network', *In 2014 23rd International Conference on Computer Communication and Networks (ICCCN), IEEE*, pp. 1-6.

[53] Amazon Elastic Load Balancing, https://aws.amazon.com/ elasticloadbalancing/

[54] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R. (2011). 'CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms', *Software: Practice and experience*, 41(1), pp. 23-50.