I.J. Modern Education and Computer Science, **2016, 3, 64-78**
Published Online March 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2016.03.08

Load Balancing in Cloud Computing: A State of the Art Survey

Mohammadreza Mesbahi
Science and Research Branch, Islamic Azad University, Tehran, Iran
E-mail: M.Mesbahi@srbiau.ac.ir

Amir Masoud Rahmani
Science and Research Branch, Islamic Azad University, Tehran, Iran
E-mail: Rahmani@srbiau.ac.ir

Abstract—Cloud computing has proposed a new perspective for provisioning the large-scale computing resources by using virtualization technology and a pay-per-use cost model. Load balancing is taken into account as a vital part for parallel and distributed systems. It helps cloud computing systems by improving the general performance, better computing resources utilization, energy consumption management, enhancing the cloud services' QoS, avoiding SLA violation and maintaining system stability through distribution, controlling and managing the system workloads. In this paper we study the necessary requirements and considerations for designing and implementing a suitable load balancer for cloud environments. In addition we represent a complete survey of current proposed cloud load balancing solutions which according to our classification, they can be classified into three categories: General Algorithm-based, Architectural-based and Artificial Intelligence-based load balancing mechanisms. Finally, we propose our evaluation of these solutions based on suitable metrics and discuss their pros and cons.

Index Terms—Cloud Computing, Load Balancing, Distributed Systems, Virtual Machine.

I. Introduction

Cloud computing is known as a popular and important term in the IT society these days. It has emerged as a large scale distributed computing paradigm that is driven by economies of scale and provides the situation that services can be dynamically configured and delivered on demand [1, 2]. To emphasize that cloud computing has some featured goals, we refer to the definition of cloud computing provided by National Institute of Standards and Technology (NIST) that says: "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models" [3, 4]. As the main goal of cloud computing we can mention the better use of distributed resources and applying them to achieve a higher throughput, performance and solving large scale computing problems [5]. Generally speaking, based on the NIST definition of cloud computing we can say best effort for offering the on demand services based on the best use of available shared resources is one of the most important goals of this model. To achieve these kinds of goals, improving the general performance of system, maintain stability, availability and some other features for a cloud computing data center, we need a mechanism which is called load balancing. Load balancing is one of the central issues and challenges in distributed systems like grid-based systems and cloud computing [6]. It is still a new problem in the cloud computing that needs new architectures and algorithms to promote the traditional approaches. In cloud computing, scheduling and handling many jobs that their arrival pattern, type of service and other properties are so hard to predict cause of the dynamic on-demand network access feature of system, an efficient load balancing mechanism is so necessary to increase the Service Level Agreement (SLA), deliver a robust service and provide other essential system requirements. In addition load balancing is an important topic because it enables other important features such as scalability [6].

Many researches have been done in the field of cloud computing and different challenges that are related which includes studies about cloud computing security and privacy like researches that have been done in [7-11] or researches related to modeling and performance analysis in cloud environments [12-16], challenges related to the frameworks and architectures of cloud [17, 18] and other general challenges in cloud computing such as scheduling, energy efficient and green computing and etc. Also there are many studies about load balancing in distributed and peer to peer networks [19-23], but a little comprehensive research about load balancing in the field of cloud computing has been done yet and we just can refer to some papers that there are in the field of load balancing in cloud computing [24-28]. In this paper we present a survey of the algorithms, architectures and all techniques which have proposed for cloud computing load balancing.

We discuss their properties and parameters that they considered and also give a comparative view of the current real load balancing mechanisms. The rest of this paper is organized as follows. In section 2 we review the literature of load balancing and will discuss the general classifications, principles and mechanism of load balancing algorithms. We survey the challenges, issues and special points that should be taken into consideration during the designing and offering a load balancer in cloud computing in section 3. In section 4 we discuss the features, positive and negative point of current cloud load balancing approaches. In addition we will have a comparative look on some parameters which each of proposed algorithms have considered. Finally, section 5 concludes paper.

II. Load Balancing Literature

Technological progress in computer world and rapid developments of distributed systems over last few years caused that load balancing problem were took into consideration as a main challenge more than ever in these systems. This section presents some important concepts and approaches of load balancing mechanism. Load balancing is the process of redistributing the general system work load among all nodes of the distributed system (network links, disk drivers, central processing units…) to improve both resource utilization and job response time while avoiding a situation where some nodes are overloaded while others are under loaded or idle [29]. Load balancing is a vital and inseparable part of cloud computing and elastic scalability [30]. In order to avoid system failure, load balancing is often used by controlling the input traffic and stop sending the workload to resources which become overloaded and non-responsive. This is an inherited feature from grid-based computing which has been transferred to cloud computing [6, 30]. Here, there are some important goals of load balancing mechanism which have been mentioned in different researches:

- Reducing job response time while keeping acceptable delays [29]
- Maintaining system stability [29]
- Having fault tolerance ability (using load balancing for implementing failover) [30]
- Improving the general system performance for achieving optimal resource utilization, maximum throughput and avoiding overload [31]
- Improving and maintaining the availability in cloud systems [32]

To start designing load balancers, there are some considerations from the point of architectural which are pointed out by [30] and we summarize them here:

- Design for providing scalability that could cover all different designing level such as CPU level, machine level, and network level or even could be at the application and data center level
- The ability of controlling the client requests and transfer to the selected resources according to the load balancer policies
- Fault tolerance for applications
- Scalability of the request handling capacity in a self-organized way
- The ability of handling more complex and higher traffic

There are many various kinds of load balancing mechanisms and approaches which most of the studies have been classified as two main categories [25, 29, 32]: static and dynamic. In static techniques [33-35], there are usually prior knowledge and some assumptions about the global status of the system such as job resource requirements, communication time, processing power of system nodes, memory and storage devices capacity and so on. A static approach is a kind of assignment from a set of tasks to a set of resources which can take either a deterministic or a probabilistic form [29]. In addition, this approach is defined usually in the design or implementation of systems [31]. In deterministic assignments, the extra workload of a certain node will be transferred to another specific node all of the time, but in probabilistic approaches each node sends its extra tasks with probability *P* to a node and with probability *1-P* to another one. The main drawback of static load balancing algorithms is that the current state of the system is not considered when making the decisions and therefore it is not a suitable approach in systems such as distributed systems which most states of the system changes dynamically. Dynamic load balancing techniques take into consideration the current state of systems that their decisions are based on. In this technique tasks can move dynamically from an overloaded node to an under-loaded one and this is the main advantage of dynamic load balancing algorithms which can change continuously according to the current state of the system. However designing and implementing a dynamic load balancing algorithm is much more complicated and harder than finding a static solution, but through dynamic mechanisms we can gain a higher performance and have more accurate and efficient solutions [25, 29]. Dynamic load balancing algorithms can be designed in two different ways: distributed and non-distributed. In distributed approaches e.g. [36-39], the load balancing process can be executed by all nodes in the system. In addition, in this approach all nodes can communicate with each other for achieving a global goal in the system which is called cooperative or every node can work independently for just achieving a local goal that is non-cooperative form. But, in a non-distributed scheme [40-43], the responsibility of balancing the system workload would not be performed by all system nodes. In centralized approach in non-distributed scheme; a single node only can execute the load balancing mechanism among all nodes. In semi-distributed form, the system will be divided into some partitions or clusters and a single node execute the load balancing process in each partition.

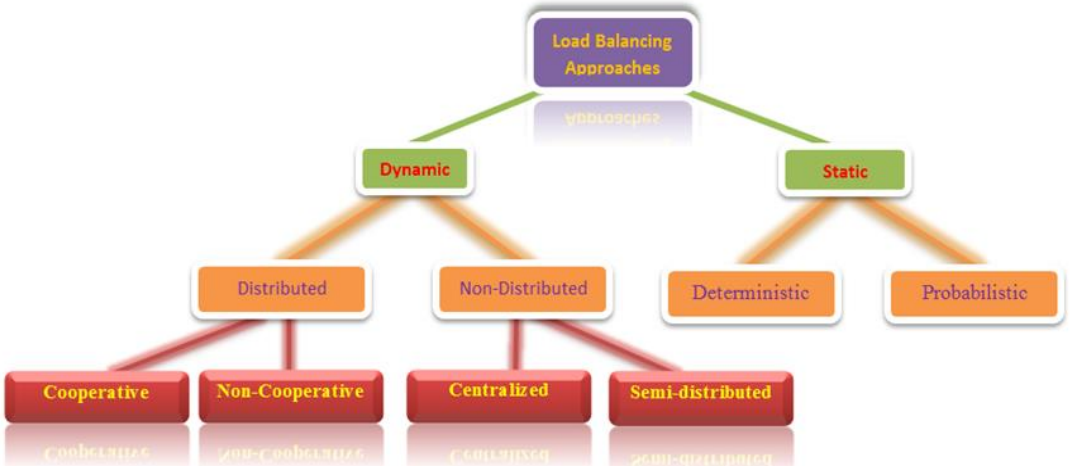

Fig.1. Load Balancing Approaches Classification

The dividing load balancing algorithms into two dynamic and static categories is based on the point that algorithms take into account the current state of system for making decision or not. But from another point of view that has mentioned in [31], load balancing techniques can be divided into two other general categories based on some other factors:

- ✓ Based on the way that the system load is distributed and resources is assigned to the tasks (Depending on the system workload)
- ✓ Based on the system topology and available information about resources

The first category is designed as:

- Centralized approach
- Distributed approach
- Mixed approach

As we discussed earlier, this classification is based on the fact that who in charge of the load distributing is. The second category is designed as:

- Static approach
- Dynamic approach
- Adaptive approach

Distributed, dynamic and adaptive load balancing mechanisms of these two categories are more suitable for large scale distributed systems such as cloud computing [31]. Adaptive approach adapts the load distribution to the system status changes, by changing their parameters dynamically and even their algorithms.

Figure 1 shows a general classification of load balancing algorithms and approaches based on common classifications and those which are proposed in [29, 31].

From the previous discussions we find out that dynamic load balancing approaches are more suitable techniques in large scale distributed systems like grid and cloud computing. For making decision based on the current workload of system, dynamic load balancers need to know some information about the state of system. Therefore a dynamic load balancing mechanism requires some components for gathering and handling these types of information. Four main components of a dynamic load balancer have been discussed in detail in [29, 44, 45] and we just introduce them here:

- Information strategy component: A component of dynamic load balancer is called information strategy component which is in charge of collecting information about status of resources in a system. According to the whole information which is obtained from each processing node in a distributed system, the load balancing process will be able to work efficiently. There are several methods that can collect system nodes' information. Some of the most popular methods are: Broadcasting, the centralized polling and agent. Information strategies that use broadcasting method [46-48], each node broadcasts its information to the system network and it will be accessible by others. In polling methods, for example [49, 50], the necessary information will be gathered by using polling techniques. An agent is one of the methods used in recent years. As some examples which use agent method for gathering information to balance the workload we can refer to [51, 52].
- Triggering Strategy component: A component of dynamic load balancer which determines the appropriate time for starting a load balancing operation is called triggering strategy component. Depending on the type of trigger policy, resources are classified into two categories: sender and receiver. Therefore, load balancing approaches will be called sender-initiated and receiver-initiated respectively [53].
- Transfer strategy component: A component which is in charge of selecting a task for transferring to another resource is called transfer strategy component. Two general approaches that determine which task should be transferred are: last-received-task and all-current-tasks. In the first approach that

is a simple solution [54, 55], a new arrived job only is selected for transferring. But in the second approach [56] an intelligent decision is made based on some parameters for selecting a job among all current nodes' jobs.

- Location strategy component: It selects a destination resource and a computing node for transferring a task. There are many approaches for selecting a destination such as: Probing, Random and Negotiation. In probing approach [54], a local node usually probing the system nodes to find a suitable destination. A location strategy with random approach selects a destination node randomly [57]. Finally in negotiation approaches, nodes negotiate with each other for load balancing process [46, 58].

Figure 2 summarizes the previous discussion and gives a general viewpoint of load balancing algorithms' components. Thus, these four components should be considered during designing a load balancer, based on special features of load balancer.

III. Cloud Load Balancer Challenges and Considerations

However cloud computing have many advances in theory and practice, researches in cloud are still in their early steps and one of the unsolved challenges is load balancing problem. Before reviewing the current load balancing approaches, we discuss the main challenges and consideration while designing cloud load balancing algorithms which are affected by cloud characteristics. The challenges are in the following points.

A. Geographically Distributed Cloud Nodes

A load balancing algorithm in cloud environment should take into account the geographically distribution of computing nodes for having an efficient performance. Some load balancing algorithms are designed for closely located nodes and therefore they have no assumption about factors such as communication delays, network bandwidth in LANS and WANs, distance between clients and computing nodes and so on [25]. Therefore, they do not work efficiently in cloud computing systems and can not satisfy the system requirements as a suitable load balancer for cloud environment. Then, for designing a cloud load balancer the spatial distribution of computing nodes must be taken into account.

B. Virtual Machines Migration

By using virtualization technique in cloud computing, a physical server can contain several virtual machines that work as independent computing nodes. A common way to unload an overloaded physical server is virtual machine migration among load balancing approaches. But there are many considerations and questions related to this migration that should be answered. During this migration we should answer to these questions [31]:

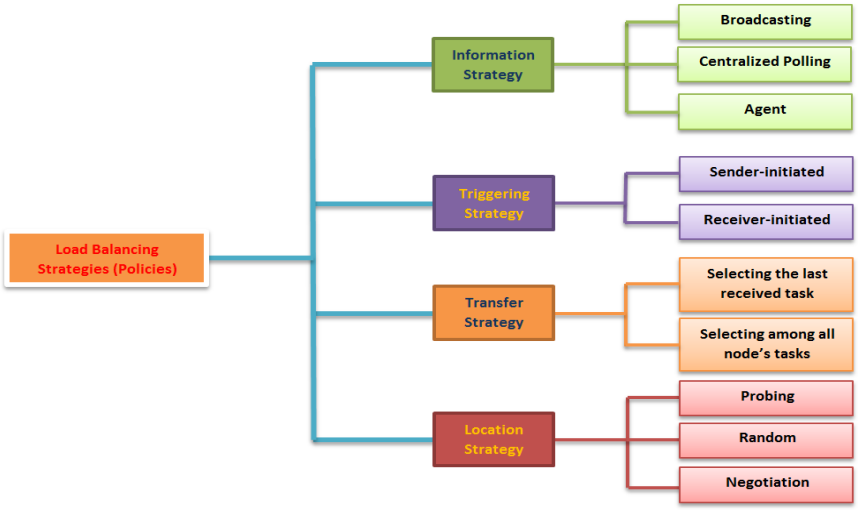

Fig.2. Four main Strategies for a load balancer

Can we guarantee the VM security when moving a virtual machine to another new location?

Can new resources (network bandwidth, computing resources…) satisfy the user requirements according to the SLA?

How can we distribute the workload dynamically during the VM migration for avoiding bottlenecks in cloud?

Therefore there are many questions and considerations that must be considered while designing a load balancer in cloud which uses VM migration technique.

C. Heterogeneous Nodes and Self-Regulating

In the past researches of load balancing, the researchers have made an assumption about homogenous nodes during a load balancing designing phase [59]. This is obviously not true in cloud computing systems where providing an on-demand access to resources or services

dynamically is a necessity and is a fundamental feature of a system. Therefore, there is a necessity to have a mechanism for finding and selecting a new instance with the same configuration as previous allocated resources to transfer the rest of client tasks and requests to new resources when the system needs to a load balancing process because of a sudden increasing of the input workload. For instance, in the Amazon EC2, dynamic load balancing is done by replicating instances of the specific middleware platform. A traffic analyzer tracks a client requests and new instances of the same platform will start when the load increases to a certain threshold [59]. Thus, in these types of solutions a combination of rules is needed for determining the conditions for load balancer. As the number of requests and active resources increase in size and complexity in cloud environment, the handling of these rules is not an easy task. In addition, these rules are static inherently and usually there is no provisioning for refinement or analysis of them [59]. Consequence of this, a load balancing mechanism should be able to self-regulates the system's workload among all cloud's computing resources according to the tasks' or clients' requirements.

D. Storage and Replication Management

Development of cloud technology has solved the problem of traditional data storage methods which require high cost of hardware and personnel management through a resource-integrated heterogeneous system that can provide the best storage, the optimal performance and the load balancing [60].

Stored data across the network have increased exponentially in the last decade for companies and individuals [31]. In this kind of situation, a full replication algorithm does not have the efficient storage utilization because of storing same data in all replication nodes in cloud computing systems. Partial replication methods can be suitable solutions for cloud-based systems which store parts of data sets in each node according to each node's computing resources like processing power and capacity [25]. But using this approach will increase the complexity of load balancing mechanisms that should be aware of availability of data sets' parts across the different computing nodes. In addition, distributing the data to the cloud for optimum data storage usage and also maintaining fast accessibility are some other related issues [25, 31]. Therefore, an efficient cloud load balancing algorithm should take into consideration the distribution of applications and their related data sets during load balancing process or VM/application migration according to the partial replication and there should be an inevitable trade-off between partial replication and discussed related concerns.

E. Load Balancer Complexity

It was concluded that load balancing algorithms which are trying to collect every detailed piece of information about system do not have a better performance considerably in comparison to algorithms which use very little or no information in information strategy phase [29]. Like this, in distributed systems and cloud computing environments, algorithms which are less complex in terms of implementation and operation are preferred and they will not have negative performance issues because of their complexity [25]. Therefore in cloud computing systems that load balancing algorithm has high complexity and requires more information, the load balancing and system's performance will be impressed by higher communication cost and delays and it is not good condition for system's efficiency. As a result of this, we can say that a suitable cloud load balancer should be designed in the simplest possible form.

F. Energy Efficiency

One of the important challenging and complicated issues in cloud computing systems is lowering the energy usage of data centers. Data centers have serious negative effects on both the environment and energy resources. Recent researches showed that large-scale data centers consumed about 1.3% of all electricity use of the world and about 2% of all electricity usage for the United States in 2010 [61]. Using internet services, computing applications and data increasing so fast which cause to have more computing resources for processing. The benefits of adoption of using cloud computing services is the economy of scale and energy saving is a key point that allows a set of global resources will be supported by reduced providers [31].

Data center resources need to be handled in an energy efficient manner therefore, a suitable load balancing algorithm for cloud systems should be able to allocate resources and redistribute the system's workloads not only to satisfy Quality of Service (QoS) which has been agreed in Service Level Agreement (SLA), but also reduce energy consumption and carbon emission [62].

G. Cloud Elasticity and Load Balancer Scalability

An important feature of cloud computing is elasticity [31]. Based on this property in cloud environments that provide the ability of resource scale up and scale down for users quickly, we can say that a load balancing algorithm in distributed systems like cloud should take into consider the system's changes in terms of size, topology,… and therefore it should be scalable, adoptable and flexible enough to allow such changes to be handled easily and also can work efficiently. In this condition, it will be able to balance the workload when computing resources and requests increase [63].

H. Load Balancer Stability and Bottleneck Prevention

As we discussed earlier, because of the dynamic nature of cloud computing systems, dynamic load balancing approaches are better choice but among dynamic load balancing algorithms, distributed dynamic load balancing algorithms are more suitable for distributed systems like cloud [25, 29]. Invoking the load balancing process and collecting information of system's node should be designed in a way that avoid having a single point of failure or bottleneck [25]. In addition, it is very important that a dynamic load balancer maintain stability in

distributed systems which one of the important factors for catching this goal is bottleneck prevention. As it has proposed in [29], a load balancing algorithm is stable if:

- Prevents the situation in which nodes spend all their time in passing jobs and applications migration instead of processing and execution.
- The workloads of two different computing nodes do not differ by more than a specific percent.
- The response time for any sudden increasing of system's workload does not exceed a specific threshold.

Therefore during designing a load balancing algorithm for cloud-based systems, stability and bottleneck prevention are two important features that should be taken into consideration.

IV. Surveying of Existing Cloud Load Balancing Approaches

In this section, we present and discuss some solutions and contributions that have been proposed in the scientific journals and conferences for cloud computing load balancing. We classify the load balancing approaches into three categories based on their perspective:

- General Algorithm-based approaches Architectural-based approaches
- Artificial intelligence-based approaches

A. General Algorithm-based Load Balancing Approaches in Cloud Computing

The General Algorithm-based load balancing approaches which we call them as "GAL-based" in an abbreviated form in this paper, are those mechanisms which propose a complete algorithm and consider all parts of load balancing algorithm within. In this category, a load balancing mechanism can be implemented or simulated based on proposed algorithm. These approaches usually propose a load balancing solution without taking into consideration special cloud architecture and represented in a general form. In other word, we can say some GAL-based load balancing mechanisms are some classical load balancing methods which are similar to the allocation and scheduling methods in operating system such as: Round-Robin, First Come First Service (FCFS), Minimum Execution Time (MET) and etc. In this section we will give a summary of the current proposed algorithms which are belongs to this category.

- Rich Lee et al. [64] proposed some most known GAL-based load balancing mechanisms and represent a good comparison of their performance with one another. We will consider and list all of them here briefly:

Round-Robin: One of the most known and the simplest algorithm for dispatching workloads to servers is round-robin that usually have good performance in systems with low workload.

Weighted round-robin: This algorithm which is derived from original round-robin has better performance compared to traditional round-robin because it assigns higher weights to servers with higher performance. Therefore, the more capable computing nodes will get more incoming workloads.

Least-connection: This algorithm counts the connections associated with each server dynamically and then based on that number it chooses the least count server and assigns new incoming workloads to the server with least connection.

Weighted least-connection: This algorithm counts the associated connections of server too, but associated new incoming workloads based on a factor that calculated by multiplying sever weight by its connection number.

Shortest expected delay: In this algorithm, the last response time for each server is considered and then the server with the least response time is selected as the next appropriate server.

- Punit Gupta et al. [65] proposed a new load balancing algorithm for better distribution of load and further enhancing the QoS. In this paper, a trust model has been presented which is based on current trust models and it uses initialization time, Machine Instruction Per Second (MIPS) and fault rate parameters to calculate trust value for each data center. In this algorithm, users and data centers are categorized into two groups: trusted and untrusted groups. According to these groups, the overloaded nodes' VMs which belong to trusted or untrusted users can migrate to trusted or untrusted data centers that are under-loaded and suitable for migrating.
- Osman Sarood et al. [66] stated that virtualization have some negative effects on HPC application. In the paper, a load balancing algorithm is proposed to achieve load balancing for tightly coupled parallel and HPC application executions in virtualized and cloud computing environments. For balancing the workloads in this proposed algorithm, processing cores is divided into two overheap (overloaded) and underSet (under-loaded) cores. Then the biggest task from most overloaded core will transfer to a suitable under-loaded core. This process is repeated until no overloaded cores are left. This load balancing algorithm is suitable for HPC applications which run iteratively and also there is interference from different VMs on the same node in their executions.
- Shu-Ching Wang et al. [67] proposed a two phases scheduling and load balancing algorithm which is presented in a three level cloud computing network. The algorithm is a combination approach of two OLB (Opportunistic Load Balancing) and LBMM (Load Balancing Min-Min). The algorithm is shown in a three level cloud computing network as follow:

In the third level, there are service nodes. Level two which consists of some service mangers is used for dividing tasks to some subtasks and assigning them to appropriate service nodes. Finally, in the first level, there is a request manager which receives the incoming workloads and sends tasks to appropriate nodes. This algorithm uses the agent-based method for gathering required information. The proposed load balancer is represented in two phases which uses the OLB algorithm for assigning tasks to service managers and in the next phase the LBMM is used to assign subtasks to the third level.

- Brototi Mandal et al. [68] proposed a load balancing algorithm that it was soft computing based. Stochastic hill climbing is a variant of hill climbing algorithm that is an incomplete approach for solving optimization problems. Because the represented load balancing algorithm is a centralized algorithm and therefore dealing with bottleneck problem, the solving optimization problem have taken into consideration for an efficient distribution of system workload.

B. Architectural-Based Load Balancing Approaches in Cloud Computing

Arch-based approaches are those mechanisms that focus on certain architecture and for achieving a load balancing, propose cloud architectures. In this category, algorithms usually not proposed explicitly and load balancing mechanism is represented through architecture components and the relations among them. Generally speaking, this solution usually is an Architectural-based solution and for catching proposed goals, a special cloud computing architecture should be taken into consideration. Here we survey some of these approaches which are proposed in our research literature.

- Monika Simjanoska et al. [69] proposed an Arch-based solution for load balancing in cloud computing. L3B is a centralized Arch-based load balancer which is placed between users and cloud heterogeneous nodes in the cloud infrastructure layer. L3B improves the overall cloud performance, reduces power consumption and customers cost through a mechanism that if the incoming workload exceeds beyond a certain threshold, a suitable VM instance will be initiated. In the other hand, if the workload decreases in compression with a certain threshold an active VM will be switched-off automatically. The proposed architecture consists of two main components: Resource Management Module (RMM) and Packet Management Module (PMM). In this architecture RMM manages the cloud resources and regularly gathers information of the active resource utilization and takes into consideration the customers' requirements. PMM module is the main core of the L3B and when the target VM responses back, PMM forwards the response packet to the client. In this paper PMM and RMM modules are discussed in detailed completely.
- GaochaoXu et al. [70] proposed a model for balancing workload in the cloud computing environment. Their solution is represented in an Arch-based which is presented through special cloud architecture. A public cloud is divided into some partitions. Partitioning helps to achieve better load balancing in the large-scale and complex environments. In the proposed solution, cloud has a main controller that communicate with the partitions balancers frequently to refresh the status information and each partition uses a local balancer that selects the best strategy for load balancing.
- Rui Wang et al. [71] addressed the load balancing problem in cloud computing through a policy for creating a load balancing algorithm in both physical servers and VMs layers. In [71] a two-level architecture is proposed and based on that, an algorithm with six general steps is designed. First of all, the statuses of nodes workload are taken into consideration by a monitor and will be stored in a database. In the next step, the nodes workload is determined according to two upper and lower bounds. Next, based on the workload changes, the future probabilistic changes will be predicated. Finally based on the gathered and evaluated information an application or a VM will be selected and application/VM migration will be started.
- Wenhong Tian et al. [72] proposed a load balancing reference architecture for load balancing and scheduling algorithm. In this paper an algorithm which is called DAIRS is presented which balances the workload in cloud computing data centers by using of three parameters: CPU, memory and network bandwidth. The proposed load balancing algorithm is considered for both physical and virtual machines. In the algorithm some parameters such as, average CPU utilization, integrated load imbalance and integrated load balance are used. In addition, four queues such as: waiting queue, requesting queue, optimizing queue and deleting queue are used in DAIRS flowchart.
- Fei Ma et al. [73] represented a model for distributed load balancing allocation of virtual machine in a cloud data center using the TOPSIS method. This method is one of the most efficient Multi Criteria Decision Making (MCDM) techniques. This paper which proposing a special architecture for balancing the load, represents a load balancing model. According to the proposed architecture, each node in a data center runs a module of the VM monitor that observes the local resource utilization. If the monitor observation reports an anomaly that resources are overloaded or under-loaded, two decisions can be made.

1. Which VM should be selected for migrating to a new PM (Physical Machine).
2. Which PM should be selected for transferring the selected VM to.

For reducing the number of probabilistic VM migrations, all VMs will be sorted on the problematic PM in decreasing order of current utilization and then the highest utilization will be selected to migrate to a new suitable PM. In this paper, the TOPSIS method is used for selecting the suitable PM.

- Jiann-Liany Chen et al. [74] state a problem that when a large number of users are accessing the cloud computing services, the computing capability of VM decreases. For solving the problem an optimal load balancing solution which is called EUQoS system for scheduling VMs is proposed. For realizing the EUQoS system, two cloud open platforms: Eucalypyus and Hadoop are used. The load balancer module in the proposed architecture consists of two components: load balancer and agent-based monitor. The load balancer module provides three mechanisms: balance triggering, EUQoS scheduling and VM control. The distribution of workload system is done by weighted round-robin load balancing algorithm.

C. Artificial Intelligence-based Load Balancing Approaches in Cloud Computing

AI-based load balancing mechanisms are the load balancing solutions that their main ideas are based on the Artificial Intelligence concepts. In the other word, AI-based approaches propose a solution for balancing the workloads in cloud computing environments through known artificial intelligence algorithms and methods by finding some similarity between them and cloud computing components and concepts. AI-based load balancing approaches can be proposed in an Arch-based form. Here we will introduce and consider some current AI-based load balancing mechanisms for cloud computing environments.

- Kumar Nishant et al. [75] proposed a modified version of Ant Colony Optimization (ACO). ACO is applied for cloud computing load balancing to help system for a proper functioning even at the peak usage hours. This AI-based solution has a head node which generates ants. Ants traverse the width and length of the cloud network in the way that they know about the location of under-loaded or over-loaded nodes. These ants along with their movements update a pheromone table for keeping the resource utilization information. In addition, movements of ants are proposed in two ways like the original ACO:

1) Forward movement
2) Backward movement

Based on the above movements and by using the pheromone table, the loads will be transferred from over-loaded nodes to under-loaded ones.

- Seyed Mohssen Ghafari et al. [76] represented a new power aware load balancing mechanism, named Bee-MMT for decreasing power consumption in cloud computing environments. This approach uses the artificial bee colony algorithm with the feature of minimal migration time. For finding over-loaded hosts the BCA is used. Then through MMT-VM selection, some VMs will be selected for migrating to a new host. In addition by using this method, after finding some under-loaded hosts, if it is possible, algorithm tries to migrate all VMs which allocated to these hosts to the other hosts while keep them not over-loaded and when all the migration have been completed, switch host to the sleep mode. Proposed algorithm includes four phases:
- 1) Host over-loading detection 2) VM selecting policy 3) Host under-loading detection 4) SLA violation metrics
- Jing Yao et al. [77] proposed an improved ABC by replacing and changing types of requests. In this paper, the similarities between ABC algorithm and cloud computing systems are discussed and because of that it is stated that this AI-based approach can be a suitable solution for solving cloud load balancing problem. The main difference of this improved ABC and original ABC is that in the proposed method, when the queue length of a physical server exceeds beyond a certain threshold, it prevents new entrants with the some type of current requests.

Here we provide a detailed investigation and analysis of reviewed load balancing mechanisms. We survey all of them one by one completely and study their objectives, main ideas, pros and cons. Table 1 gives brief information of this study.

In addition, for a load balancing algorithm some features and consideration should be taken into consideration as we mentioned earlier. Some various parameters are proposed and defined in [78] such as response time, scalability, throughput, resource utilization, fault tolerance, migration time and etc. We survey some of them and some other challenges and considerations which were mentioned earlier in section 3, for the surveyed load balancing algorithms here. The table 4 shows the results of this study.

In this table we categorized some current load balancing mechanism for cloud computing environments that are proposed in scientific journals and conferences in recent years. As mentioned earlier, according to our classification these solutions are classified into three GAL-based, Arch-based and AI-based load balancing mechanisms. We considered all these methods from different aspects. The first point is related to dynamic or static classifications. Because all surveyed load balancing methods have been proposed for cloud computing environments. Therefore all of them have applied dynamic approach which is suitable for systems like cloud computing that have dynamic, unpredictable and various workloads. In addition, most of the proposed load

balancing mechanisms which we considered, have been designed in non-distributed forms, it means centralized and semi-distributed. So, most of them are suffering from the bottleneck problem. As a result of this issue, we can say that all of them have a single point of failure and therefore they are not fault tolerant solutions and this would be a main drawback for most of them. As another aspect, we mentioned in table 4 that all of these distributed and non-distributed approaches in this study are scalable.

Table 1. Overview of Current GAL-Based Load Balancing Techniques

Load Balancing Techniques	Main Idea	Primary Objectives	Pros	Cons
Trust Management - 2013 [65]	• Proposing a trust module for trust value management for the cloud IaaS parameters	• Increasing the resource utilization • Decreasing the request response time of system	• Taking into consideration the level of trust in both load balancing process and VM migration • Enhancement of QoS based on the proposed model	• does not provide the code or pseudo-code of proposed solution • does not provide the proof of the trust formula • No comparison of the proposed method with other similar techniques • does not provide a formal description of the expressed model
Cloud Friendly Load Balancing - 2012 [66]	• Using object migration for achieving load balancing for tightly coupled parallel applications executing in a virtualized environment that suffers from interfering jobs	• Reducing the energy consumption and timing penalty • Decreasing the execution time for a virtualized environment	• Reducing the jobs execution time through proposed load balancing algorithm • Reducing the power and energy consumption	• The main core runs the load balancing process which causes a bottleneck and therefore a single point of failure in the algorithm • The lack of fault tolerant • does not determine the appropriate destination node
Two-Phase Load Balancing - 2010 [67]	• Combining the OLB (opportunistic load balancing) and LBMM (load balance Min-Min) scheduling algorithms for utilizing better executing efficiency	• Maintaining the load balancing of the system • Minimizing the execution time • Utilizing better executing efficiency	• Using of the both OLB and LBMM in the different architecture's levels • Using different service thresholds Such as: remaining CPU, remaining memory and transmission rate	• Lack of simulation or implementation of the proposed algorithm • does not provide the performance evaluation metrics • Request manager is a bottleneck (a single point of failure of the algorithm) • No comparison of proposed algorithm with other similar methods
Stochastic Hill Climbing (SCH) - 2012 [68]	• Proposing a load balancing mechanism using soft computing approaches and a local optimization of stochastic hill climbing	• Load distribution in cloud computing	• Comparison with the other algorithm like round-robin and FCFS • Better performance in comparison with FCFS and round-robin	• The algorithm is completely centralized and runs by a single mode which causes a bottleneck and a single point of failure • does not provide the performance evaluation metrics • No attention to current system load • Lack of resource utilization

As another metric for evaluating proposed cloud load balancing approaches we referred to algorithm complexity problem in section 3. In table 4, we cover this aspect through an overhead parameter. We describe the proposed load balancing algorithm by three qualitative statements: low, medium and high which are indicating the overhead level for implementing the proposed solution.

Quality of service has special place in cloud services in related discussions. Cloud computing systems offer their services based on pay-as-you-go cost model. Therefore, providing the proposed quality in SLA for customers correctly is very important. As one of the main goals of cloud load balancers, QoS should be taken into account for offering the best suitable required quality of services during balancing the system workload and assigning the jobs to resources. The results of our study in table 4 show that most load balancing mechanisms that have taken into account the SLA violation metrics as one of the evaluation metrics have been considered the QoS problem during their load balancing processes.

One of the other important considerations is energy efficiency and energy consumption problem in cloud computing. In table 4 we determined that which load balancing mechanisms have taken into account this point while they offer their load balancers. The trend of recent researches shows that the energy efficiency becomes an important consideration during designing load balancing mechanisms for cloud computing in recent years.

A common solution in almost all cloud load balancing mechanisms is application or VM migration that some jobs will be transferred from overloaded nodes to under-loaded ones for gaining a normal state in the system. In this situation, the migration time can be counted as one of the main factors for evaluating load balancing approaches in cloud systems. Therefore, those solutions that have considered the time migration in their load balancers and using some mechanisms for reducing the migration time and therefore reducing the service response time can be more successful than other approaches those using migration techniques without considering the migration time. In table 4 we determined those solutions that used migration time factor in their load balancers.

Finally we surveyed the different evaluation metrics and tools which have been used for evaluating these proposed load balancers. Simulation is a very good and reasonable way for evaluating proposed mechanisms by simulating the real world conditions and situations. Most

of these surveyed load balancing approaches have used the simulation for performance evaluating of their mechanisms instead of implementing their load balancing solutions in a real cloud environment. They use simulators like CloudSim Toolkit [79] and CloudAnalyst [80] which are popular and common cloud computing environments simulator as we mentioned in table 4.

Table 2. Overview of Current Arch-Based Load Balancing Techniques

Load Balancing Techniques	Main Idea	Primary Objectives	Pros	Cons
3B - 2013 [69]	• Proposing a centralized load balancer (L3B) between users and heterogeneous nodes of cloud in the infrastructure level	• Improving the cloud computing performance • Reducing the power consumption • Reducing the customers' costs	• L3B activates and deactivates the cloud resources that reduces costs for consumers • Maximizing resource utilization and reducing power consumption • High elasticity • Reducing energy consumption	• Computational overhead is very high for the proposed architecture • The high latency in L3B modules • No simulation or implementation of the proposed solution • No comparison of L3B with other methods • Centralized approach causes a bottleneck in the system architecture which is a single point of failure
Cloud Partition Load Balancing- 2013 [70]	• Representing a load balancing model based on cloud partitioning with a switching mechanism for selecting different strategies	• Improving the efficiency in the public cloud environment • Proposing a load balancing model for public cloud computing to improve utilization	• Using algorithms with low complexity for under-loaded situations in partitions • Using different load balancing strategies for each partition based on different special situations	• No simulation or implementation of proposed solution • does not provide any algorithm code or pseudo-code • No mechanism for determining upper and lower bounds • No performance evaluation metrics • Main controller is a bottleneck (a single point of failure)
VM-based Two Dimensioned Load Management – 2011 [71]	• Using prediction and estimation methods for its model in order to decide about VM migration or job migration	• Balancing the workload in both VMs and PMs levels • Selecting the best VM from all VMs for migration • Deciding for selecting job migration or VM migration based on the system load	• Using both VM migration and application migration in the cloud system based on the current situation • Using load estimation method for avoiding the unnecessary migrations according to the future workload • Reducing system overhead by reducing migration	• Considering only applications with seasonal attribute changes which is not extendable for a real system • No method for determining upper and lower bounds • does not provide any code or pseudo code of proposed solution • does not provide performance evaluation • No resource utilization • A single scheduler is a bottleneck and a single point of failure
DAIRS - 2011 [72]	• Representing a load balancing algorithm based on all parameters: CPU, memory and network bandwidth loads for both VMs and PMs	• Balancing the load through an integrated load balancing scheduling algorithm in cloud data centers	• Comparisons of proposed load balancing mechanism with some similar methods • Using all CPU, memory and network bandwidth loads	• One single scheduler is a bottleneck in the algorithm and a single point of failure • Misusing of the parameters in the proposed formula • does not provide any code or pseudo code of proposed mechanism • The proposed load balancing approach is completely centralized which is not suitable for cloud systems
TOPSIS Method – 2012 [73]	• Proposing a new model for distributed load balancing allocation of virtual machine in cloud data center using the TOPSIS method	• Achieving better load balancing in a large-scale cloud computing data center • Reducing the VM migration rate	• Using of TOPSIS Method as one of the most efficient MCDM technique • Switching off the under loaded physical machine by migrating their VMs • Considering all CPU usage, memory usage and bandwidth usage as decision parameters	• does not provide any code or pseudo code of proposed method • No parameters for determining upper and lower bounds • Heavy computational overhead because of using TOPSIS method for selecting suitable PM • Limiting the QoS parameters to only requested MIPS criterion in SLA violation
EuQoS – 2012 [74]	• Extending EUQOS to accommodate real-time services • Using log processing services for investigating the system performance	• Improving the performance of VMs in Eucalyptus • Proposing a load balancing method which can guarantee QoS	• Representing a good architecture in details with all component • Practical comparison of proposed load balancing solution with Hadoop and Eucalyptus cloud frameworks	• Using a complete centralized method which can be considered as a bottleneck and a single point of failure • Simulating the proposed method with a small number of virtual machines for load balancing proce • Using the weighted round-robin instead of proposing a load balancing algorithm • No assurance for providing the promised QoS

Table 3. Overview of Current AI-Based Load Balancing Techniques

Load Balancing Techniques	Main Idea	Primary Objectives	Pros	Cons
Ant Colony Optimization (ACO) – 2012 [75]	• Balancing the workload in cloud computing systems using Ant Colony Optimization (ACO)	• Distribution of the load among the cloud nodes efficiently • Building an optimum solution set	• Existing a single pheromone table which can be updated by all ants and improving gradually • Using a timer for each ant for committing suicide • Good detection mechanism of overloaded and under-loaded nodes	• Head node in proposed algorithm is a bottleneck • does not provide a measurement technique for calculating the number of required ants • High computational overhead for calculating pheromone formulas • does not provide any simulation or implementation of the proposed algorithm • Occupied bandwidth due to the presence of the ants • Single point of failure
Bee-MMT – 2013 [76]	• Using Bee colony for balancing the workload in cloud computing using VM migration	• Decreasing power consumption • Decreasing the CO2 emission and operational cost	• Using VM migration by applying minimum migration time method • Reducing energy consumption using VM migration • Considering cloud QoS • Comparison of proposed solution with other similar methods	• Generating bees from a single resource can be considered as a bottleneck • High complexity of algorithm for selecting best overloaded host • No prediction for future workload of hosts
Improved ABC - 2012 [77]	• Improving the original ABC based on the types of requests	• Proposing a load balancing strategy for cloud computing systems by artificial Bee algorithm	• Improving the system throughput • Simulating the proposed algorithm • Comparing proposed algorithm with other similar methods	• does not provide any code or pseudo-code of improved ABC • High computational overhead for implementing the improved ABC • Lack of suitable performance of the proposed method in under loaded situations • No resource utilization • Lower stability in comparison with original ABC

V. Conclusion and Future Works

Load balancing in cloud computing data centers has been a main challenge and an active area of research in recent years. In this paper we have presented a survey on current load balancing techniques and solutions which have proposed only for cloud computing environments. According to our study, cloud load balancing mechanism can be categorized into three main groups based on their designing perspectives: General Algorithm-based, Architectural-based and Artificial Intelligence-based load balancing approaches. In addition, our survey on the current proposed cloud load balancing approaches has some contributions which we list the main ones here:

1) A new Classification of cloud load balancing in cloud computing environment based on the designing perspective has been proposed in this paper.
2) According to our study results the best load blanking approaches in cloud computing are those ones which have dynamic, distributed and non-cooperative features. Dynamic feature is referred to dynamic and on demand property in cloud which is resulted in dynamic workload and services. Distributed designing can prevent the bottleneck and a single point of failure and therefore provides the better scalability and fault tolerant abilities. Finally non-cooperative approach can avoid unnecessary overheads while a load balancer is performing.
3) There are many architectural and designing time considerations for implementing a cloud load balancer that should be taken into account such as: virtual machine migration, elasticity, cost and energy consumption management which we proposed a complete discussion in section 3.
4) The result of this study in table 2 shows that the most of the proposed load balancing solutions are suffering from the bottleneck and single point of failure problems in the first place.
5) The recent cloud load balancing solutions are focusing on Green cloud topic in load balancing mechanisms by considering the challenges like: Reducing the energy and power consumption, Reducing carbon emission and also reducing cost customers as a result of energy efficiency topic.

As the future work of this paper, we will consider more cloud load balancing solutions according to our three levels classification and survey the load balancing solutions' trend.

In addition it will be possible to evaluate the current load balancing mechanisms through simulating and taking into consideration more metrics for their evaluations.

Table 4. Metrics Considered By Surveyed Load Balancing Mechanisms in Cloud Computing

Techniques	Category	Non Dist./Distributed	Dynamic / Static	Evaluation Metrics	Overhead	Fault Tolerant	QoS	Scalability	Energy Efficient	Migration Time	Simulation/ Implementation
LB1 [65]	GAL-based	--------	Dynamic	--------	low	×	×	✓	×	×	Implementation /using Qemu and Libvirt
LB2 [66]	GAL-based	Non Distributed	Dynamic	-Power consumption -Execution time	low	×	×	✓	✓	✓	Implementation In test bed
LB3 [67]	GAL-based	Non Distributed	Dynamic	Execution time	low	×	×	✓	×	×	--------
LB4 [68]	GAL-based	Non Distributed	Dynamic	Performance	low	×	×	✓	×	×	Simulation Cloud analyst
LB5 [69]	Arch-based	Non Distributed	Dynamic	Response time	Medium	×	×	✓	✓	×	--------
LB6 [70]	Arch-based	Non Distributed	Dynamic	--------	low	×	×	✓	×	×	--------
LB7 [71]	Arch-based	Non Distributed	Dynamic	-VMs load -PMs load	low	×	✓	✓	×	×	Simulation
LB8 [72]	Arch-based	Non Distributed	Dynamic	Running time -Average Imbalance Value	Medium	×	×	✓	×	×	Simulation
LB9 [73]	Arch-based	Distributed	Dynamic	-SLA violation -Number of VM Migration -Number of Used PMs	High	✓	✓	✓	✓	✓	Simulation CloudSim Toolkit
LB10 [74]	Arch-based	Non Distributed	Dynamic	-Throughput -Execution time	low	×	✓	✓	×	×	Emulation (Eucalyptus, Hadoop
LB11 [75]	AI-based	--------	Dynamic	--------	High	×	×	✓	×	×	--------
LB12 [76]	AI-based	Distributed	Dynamic	-Power consumption -SLA Violation -Number of Migration	High	✓	✓	✓	✓	✓	Simulation CloudSim Toolkit
LB13 [77]	AI-based	--------	Dynamic	Throughput	High	×	×	✓	×	×	Simulation Repast.Net

REFERENCES

[1] Foster, I., et al. *Cloud computing and grid computing 360-degree compared*. in *Grid Computing Environments Workshop, 2008. GCE'08*. 2008. Ieee.

[2] Rastogi, G. and R. Sushil, *Cloud Computing Implementation: Key Issues and Solutions*, in *2nd International Conference on Computing for Sustainable Global Development (INDIACom)*2015, IEEE. p. 320-324.

[3] Lee Badger, T.G., Robert Patt, Corner JeffVoas. *DRAFT Cloud Computing Synopsis and Recommendations*. 2011; Available from: http://csrc.nist.gov/publications/nistpubs/800-146/sp800-146.pdf.

[4] Puthal, D., et al., *Cloud Computing Features, Issues, and Challenges: A Big Picture*, in *International Conference on Computational Intelligence and Networks (CINE)*2015, IEEE. p. 116-123.

[5] Jadeja, Y. and K. Modi. *Cloud computing-concepts, architecture and challenges*. in *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*. 2012. IEEE.

[6] Rimal, B.P., E. Choi, and I. Lumb. *A taxonomy and survey of cloud computing systems*. in *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*. 2009. Ieee.

[7] Zissis, D. and D. Lekkas, *Addressing cloud computing security issues.* Future Generation Computer Systems, 2012. **28**(3): p. 583-592.

[8] Subashini, S. and V. Kavitha, *A survey on security issues in service delivery models of cloud computing.* Journal of Network and Computer Applications, 2011. **34**(1): p. 1-11.
[9] Khan, A.N., et al., *Towards secure mobile cloud computing: A survey.* Future Generation Computer Systems, 2012.
[10] 1Lombardi, F. and R. Di Pietro, *Secure virtualization for cloud computing.* Journal of Network and Computer Applications, 2011. **34**(4): p. 1113-1122.
[11] Khorshed, M.T., A. Ali, and S.A. Wasimi, *A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing.* Future Generation Computer Systems, 2012. **28**(6): p. 833-851.
[12] Mauch, V., M. Kunze, and M. Hillenbrand, *High performance cloud computing.* Future Generation Computer Systems, 2012.
[13] Ghosh, R., et al., *Modeling and performance analysis of large scale IaaS Clouds.* Future Generation Computer Systems, 2012.
[14] Garg, S.K., S. Versteeg, and R. Buyya, *A framework for ranking of cloud computing services.* Future Generation Computer Systems, 2013. **29**(4): p. 1012-1023.
[15] Pérez-Miguel, C., A. Mendiburu, and J. Miguel-Alonso, *Modeling the availability of Cassandra.* Journal of Parallel and Distributed Computing, 2015.
[16] Sousa, E., et al., *A Modeling Approach for Cloud Infrastructure Planning Considering Dependability and Cost Requirements.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, , 2015. **45**(4): p. 549-558.
[17] Yang, X., et al., *A business-oriented Cloud federation model for real-time applications.* Future Generation Computer Systems, 2012. **28**(8): p. 1158-1167.
[18] Dukaric, R. and M.B. Juric, *Towards a unified taxonomy and architecture of cloud frameworks.* Future Generation Computer Systems, 2012.
[19] Fu, S., C.-Z. Xu, and H. Shen, *Randomized load balancing strategies with churn resilience in peer-to-peer networks.* Journal of Network and Computer Applications, 2011. **34**(1): p. 252-261.
[20] Wu, D., Y. Tian, and K.-W. Ng, *Resilient and efficient load balancing in distributed hash tables.* Journal of Network and Computer Applications, 2009. **32**(1): p. 45-60.
[21] Raj, J.S. and R. Fiona. *Load balancing techniques in grid environment: A survey*. in *Computer Communication and Informatics (ICCCI), 2013 International Conference on*. 2013. IEEE.
[22] Randles, M., et al. *A Comparative Experiment in Distributed Load Balancing*. in *Developments in eSystems Engineering (DESE), 2009 Second International Conference on*. 2009. IEEE.
[23] Khan, R.Z. and M.F. Ali, *An Efficient Diffusion Load Balancing Algorithm in Distributed System.* International Journal of Information Technology and Computer Science (IJITCS), 2014. **6**(8): p. 65.
[24] Mohamed, N., J. Al-Jaroodi, and A. Eid, *A Dual-Direction Technique for Fast File Downloads with Dynamic Load Balancing in the Cloud.* Journal of Network and Computer Applications, 2013.
[25] Nuaimi, K.A., et al. *A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms*. in *Network Cloud Computing and Applications (NCCA), 2012 Second Symposium on*. 2012. IEEE.
[26] Suresh, M., K.B. Santhosh, and S. Karthik. *A Load Balancing Model in Public Cloud Using ANFIS and GSO*. in *Intelligent Computing Applications (ICICA), 2014 International Conference on*. 2014. IEEE.
[27] Kanakala, R. and V.K. Reddy, *Performance Analysis of Load Balancing Techniques in Cloud Computing Environment.* TELKOMNIKA Indonesian Journal of Electrical Engineering, 2015. **13**(3): p. 568-573.
[28] Poddar, R., A. Vishnoi, and V. Mann, *HAVEN: Holistic Load Balancing and Auto Scaling in the Cloud*, in *7th International Conference on Communication Systems and Networks (COMSNETS)*2015.
[29] Alakeel, A.M., *A guide to dynamic load balancing in distributed computer systems.* International Journal of Computer Science and Information Security, 2010. **10**(6): p. 153-160.
[30] Rimal, B.P., et al., *Architectural requirements for cloud computing systems: an enterprise cloud approach.* Journal of Grid Computing, 2011. **9**(1): p. 3-26.
[31] Khiyaita, A., et al. *Load balancing cloud computing: state of art*. in *Network Security and Systems (JNS2), 2012 National Days of*. 2012. IEEE.
[32] Zenon, C., M. Venkatesh, and A. Shahrzad, *Availability and Load Balancing in Cloud Computing.* 2011.
[33] Alexeev, Y., et al. *Heuristic static load-balancing algorithm applied to the fragment molecular orbital method*. in *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*. 2012. IEEE.
[34] Chonggun, K. and H. Kameda, *Optimal static load balancing of multi-class jobs in a distributed computer system.* IEICE TRANSACTIONS (1976-1990), 1990. **73**(7): p. 1207-1214.
[35] Penmatsa, S. and A.T. Chronopoulos, *Game-theoretic static load balancing for distributed systems.* Journal of Parallel and Distributed Computing, 2011. **71**(4): p. 537-555.
[36] Cosenza, B., et al. *Distributed load balancing for parallel agent-based simulations*. in *Parallel, Distributed and Network-Based Processing (PDP), 2011 19th Euromicro International Conference on*. 2011. IEEE.
[37] Shi, J., C. Meng, and L. Ma. *The Strategy of Distributed Load Balancing Based on Hybrid Scheduling*. in *Computational Sciences and Optimization (CSO), 2011 Fourth International Joint Conference on*. 2011. IEEE.
[38] Gonzalez-Ruiz, A. and Y. Mostofi. *Distributed load balancing over directed network topologies*. in *American Control Conference, 2009. ACC'09.* 2009. IEEE.
[39] Riakiotakis, I., et al., *Distributed dynamic load balancing for pipelined computations on heterogeneous systems.* Parallel Computing, 2011. **37**(10): p. 713-729.
[40] Ahmad, I. and A. Ghafoor. *A semi distributed load balancing scheme for large multicomputer systems*. in *Parallel and Distributed Processing, 1990. Proceedings of the Second IEEE Symposium on*. 1990. IEEE.
[41] Ahmad, I. and A. Ghafoor, *Semi-distributed load balancing for massively parallel multicomputer systems.* Software Engineering, IEEE Transactions on, 1991. **17**(10): p. 987-1004.
[42] Zhu, W., C. Sun, and C. Shieh. *Comparing the performance differences between centralized load balancing methods*. in *Systems, Man, and Cybernetics, 1996., IEEE International Conference on*. 1996. IEEE.
[43] Das, S., H. Viswanathan, and G. Rittenhouse. *Dynamic load balancing through coordinated scheduling in packet data systems*. in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*. 2003. IEEE.
[44] Patni, J.C., et al. *Load balancing strategies for Grid computing*. in *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*. 2011. IEEE.

[45] Rajavel, R. *De-Centralized Load Balancing for the Computational Grid environment*. in *Communication and Computational Intelligence (INCOCCI), 2010 International Conference on*. 2010. IEEE.

[46] Ni, L.M., C.-W. Xu, and T.B. Gendreau, *A distributed drafting algorithm for load balancing.* Software Engineering, IEEE Transactions on, 1985(10): p. 1153-1161.

[47] Baumgartner, K.M. and B.W. Wah. *A global load balancing strategy for a distributed computer system*. in *Distributed Computing Systems in the 1990s, 1988. Proceedings., Workshop on the Future Trends of*. 1988. IEEE.

[48] Felber, P., et al., *Survey on Load Balancing in Peer-to-Peer Distributed Hash Tables.*

[49] Dinan, J., et al. *Dynamic load balancing of unbalanced computations using message passing*. in *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*. 2007. IEEE.

[50] Lee, G.-H. *Using system state information for adaptive state polling policy in distributed load balancing*. in *Parallel Algorithms/Architecture Synthesis, 1997. Proceedings. Second Aizu International Symposium*. 1997. IEEE.

[51] Brazier, F., et al. *Agents negotiating for load balancing of electricity use*. in *Distributed Computing Systems, 1998. Proceedings. 18th International Conference on*. 1998. IEEE.

[52] Cao, J., et al. *Agent-based grid load balancing using performance-driven task scheduling*. in *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*. 2003. IEEE.

[53] Eager, D.L., E.D. Lazowska, and J. Zahorjan, *A comparison of receiver-initiated and sender-initiated adaptive load sharing.* Performance evaluation, 1986. **6**(1): p. 53-68.

[54] Eager, D.L., E.D. Lazowska, and J. Zahorjan, *Adaptive load sharing in homogeneous distributed systems.* Software Engineering, IEEE Transactions on, 1986(5): p. 662-675.

[55] Evans, D. and W. Butt, *Dynamic load balancing using task-transfer probabilities.* Parallel Computing, 1993. **19**(8): p. 897-916.

[56] Goswami, K.K., M. Devarakonda, and R.K. Iyer, *Prediction-based dynamic load-sharing heuristics.* Parallel and Distributed Systems, IEEE Transactions on, 1993. **4**(6): p. 638-648.

[57] Yang, C.-C., C. Chen, and J.-Y. Chen. *Random early detection web servers for dynamic load balancing*. in *Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on*. 2009. IEEE.

[58] Chechina, N., P. King, and P. Trinder. *Using negotiation to reduce redundant autonomous mobile program movements*. in *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*. 2010. IEEE.

[59] Randles, M., D. Lamb, and A. Taleb-Bendiab. *A comparative study into distributed load balancing algorithms for cloud computing*. in *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on*. 2010. IEEE.

[60] Wu, T.-Y., et al. *Dynamic load balancing mechanism based on cloud storage*. in *Computing, Communications and Applications Conference (ComComAp), 2012*. 2012. IEEE.

[61] Rahman, A., X. Liu, and F. Kong, *A Survey on Geographic Load Balancing Based Data Center Power Management in the Smart Grid Environment.*

[62] Beloglazov, A., J. Abawajy, and R. Buyya, *Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing.* Future Generation Computer Systems, 2012. **28**(5): p. 755-768.

[63] Ray, S. and A. De Sarkar, *EXECUTION ANALYSIS OF LOAD BALANCING ALGORITHMS IN CLOUD COMPUTING ENVIRONMENT.* International Journal, 2012.

[64] Lee, R. and B. Jeng. *Load-balancing tactics in cloud*. in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on*. 2011. IEEE.

[65] Gupta, P., M.K. Goyal, and P. Kumar. *Trust and reliability based load balancing algorithm for cloud IaaS*. in *Advance Computing Conference (IACC), 2013 IEEE 3rd International*. 2013. IEEE.

[66] Sarood, O., A. Gupta, and L.V. Kalé. *Cloud Friendly Load Balancing for HPC Applications: Preliminary Work*. in *Parallel Processing Workshops (ICPPW), 2012 41st International Conference on*. 2012. IEEE.

[67] Wang, S.-C., et al. *Towards a load balancing in a three-level cloud computing network.* in *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*. 2010. IEEE.

[68] Mondal, B., K. Dasgupta, and P. Dutta, *Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach.* Procedia Technology, 2012. **4**: p. 783-789.

[69] Simjanoska, M., et al. *L3B: Low level load balancer in the cloud*. in *EUROCON, 2013 IEEE*. 2013. IEEE.

[70] Xu, G., J. Pang, and X. Fu, *A load balancing model based on cloud partitioning for the public cloud.* Tsinghua Science and Technology, 2013. **18**(1): p. 34-39.

[71] Wang, R., W. Le, and X. Zhang, *Design and implementation of an efficient load-balancing method for virtual machine cluster based on cloud service.* 2011.

[72] Tian, W., et al. *A dynamic and integrated load-balancing scheduling algorithm for Cloud datacenters*. in *Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on*. 2011. IEEE.

[73] Ma, F., F. Liu, and Z. Liu. *Distributed load balancing allocation of virtual machine in cloud data center*. in *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on*. 2012. IEEE.

[74] Chen, J.-L., Y.T. Larosa, and P.-J. Yang. *Optimal QoS load balancing mechanism for virtual machines scheduling in Eucalyptus cloud computing platform*. in *Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on*. 2012. IEEE.

[75] Nishant, K., et al. *Load Balancing of Nodes in Cloud Using Ant Colony Optimization*. in *Computer Modelling and Simulation (UKSim), 2012 UKSim 14th International Conference on*. 2012. IEEE.

[76] Ghafari, S.M., et al. *Bee-MMT: A load balancing method for power consumption management in cloud computing*. in *Contemporary Computing (IC3), 2013 Sixth International Conference on*. 2013. IEEE.

[77] Yao, J. and J.-h. He. *Load balancing strategy of cloud computing based on artificial bee algorithm*. in *Computing Technology and Information Management (ICCM), 2012 8th International Conference on*. 2012. IEEE.

[78] Kansal, N.J. and I. Chana, *Cloud Load Balancing Techniques: A Step Towards Green Computing.* IJCSI International Journal of Computer Science Issues, 2012. **9**(1): p. 238-246.

[79] Calheiros, R.N., et al., *CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms.* Software: Practice and Experience, 2011. **41**(1): p. 23-50.
[80] Wickremasinghe, B., R.N. Calheiros, and R. Buyya. *Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications*. in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*. 2010. IEEE.

Authors' Profiles

Mohammad Reza Mesbahi received his B.Sc. in computer engineering from Islamic Azad University, Hamedan, in 2011. He received his M.Sc. in computer engineering from Science and Research Branch, Islamic Azad University of Tehran, Iran, in 2013. He is now a Ph.D. student at IAU University. His major interests are Distributed Systems, Grid and Cloud Computing, Software Architecture and Big Data.

Amir Masoud Rahmani received his B.S. in computer engineering from Amir Kabir University, Tehran, in 1996, the M.S. in computer engineering from Sharif University of technology, Tehran, in 1998 and the PhD degree in computer engineering from IAU University, Tehran, in 2005. He is associate professor in the Department of Computer Engineering at the IAU University. He is the author/co-author of more than 140 publications in technical journals and conferences. He served on the program committees of several national and international conferences. His research interests are in the areas of distributed systems, ad hoc and sensor wireless networks, scheduling algorithms and evolutionary computing.

