International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||

A Machine Learning Technique Based Survey to Perform Load Balancing in Cloud Computing

Shuchita Mudgil[1], Rohan Rajoriya[2], Shiv Kumar Tiwari[3]

Lecturer, Department of Information Technology, Kalaniketan Polytechnic College, Jabalpur, MP India[1, 2]

Research Scholar, Department of Computer Science & Engineering, Amity University, Gwalior, MP, India[3]

ABSTRACT: Cloud computing is currently becoming increasingly popular because of its diverse characteristics such as its accessibility, low cost and often low power consumption. Many algorithms and techniques have been proposed for scheduling virtual machines to provide dynamic load balancing, dynamic scalability, and resource reallocation. Intelligent algorithms are used to optimize results and minimize makepan scheduling while using dynamic environment-based resources efficiently. This paper examines various intelligent scheduling algorithms such as Genetic Algorithm (GA), Simulated Annealing (SA), Tabu Search (TS), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Immune System (AIS), Bacterial Foraging Algorithm (BF), Fish Swarm Optimization Algorithm (FS), Cat Swarm Optimization Algorithm (CS), Firefly Algorithm (FF), Cuckoo Search Algorithm (CS).
This paper also compares cloud computing scheduling with respect to makepan, load balancing, CPU utilization, time limit, response time, and allocation costs. In addition, the paper proposes an abstract model to integrate the desirable features of a cloud-friendly algorithm. Future research opportunities and the end of the paper are highlighted.

KEYWORDS: Cloud Computing, Scheduling Algorithms, Model, Survey

I. INTRODUCTION

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

A cloud computing model is effective if its services are used in the best possible way and utilizing and maintaining careful management of cloud resources will achieve such an effective use. Managing resources is accomplished by the implementation of rigorous resource management, allocation and efficient resource scalability techniques. Through a process known as virtualization which makes use of an entity (software, hardware or both) known as hypervisor[8], these services are provided to customers in the form of Virtual Machines (VM). The greatest benefit of cloud computing is that a single physical user computer is turned into a virtual multiuser machine[9, 10]. The Cloud Service Provider (CSP) plays a vital role in delivering service to customers, which is a complex task with virtual resources available. Some VMs will get a heavy traffic of user tasks when fulfilling user requests and some will get a lesser traffic. As a result, the Cloud Service Provider (CSP) is left with unbalanced machines that have an immense gradient of user tasks and usage of resources[11].

The issue of load unbalancing is an undesirable occurrence on the CSP side that degrades the efficiency and effectiveness of

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||

the computing resources along with guaranteed quality of service (QoS) on the agreed service level agreement (SLA) between the customer and the provider. The need for load balancing (LB) emerges under these circumstances, and is a peculiar topic of research interest among researchers. Cloud computing load balancing may be performed at level of the physical computer or VM[2].

A task requires resources from a VM and when a bunch of tasks arrive at a VM, the resources are depleted which means that no resource is now available to handle the additional requests for tasks. The VM is said to have entered a surcharged state when such a condition occurs. At this point in time, tasks will either suffer from hunger or end up in impasse with no chance of accomplishing them. Consequently tasks on other VM need to be migrated to another tool. The method of workload migration involves three basic steps: load balancing that tests the current load on system resource, resource exploration that finds another appropriate resource and workload migration that transfers extra tasks to the available resources. Three separate units commonly known as load balancer, resource discovery and task migration units perform these operations, respectively.

Load balancing is the method of redistributing workload in a distributed network such as cloud computing ensuring that no computer machine is overloaded, underloaded or idle[12, 13]. Load balancing attempts to speed up various restricted parameters such as response time, execution time, device reliability etc. thus enhancing cloud performance [14, 15]. It is a technique of optimization, in which scheduling of tasks is a difficult NP problem. Researchers have suggested a large range of load balancing strategies where the emphasis was mainly on task scheduling, task allocation, resource scheduling, resource allocation and resource management. To the best of our knowledge, we have not been able to find an in-depth and detailed literature about factors that cause unbalancing of the load situation. Load balancing based survey papers did not include a clear systematic method and methodology classification. The paper's main goal is to review the current research along with the advantages and disadvantages in it. A contrast is also made between different current load balancing strategies and the cloud load balancing problems facing them. The survey also discusses reasons responsible for the question of load unbalancing and recommends approaches that could be used in future wo rk. This paper's contributions are summarised as follows:

I.Explore factors in cloud computing that cause unbalanced load problems.

II.Provide a detailed overview of the current methods in the field of load balancing and how those methods were used in cloud technology.

III.Provide a detailed overview of various load balance techniques , processes, approaches, and algorithms.

IV. Analyze the challenges faced by researchers in developing an efficient load balancing algorithm.

The remaining paper is structured as follows. Section “Load balancing model background” features a brief description about load balancing model in cloud computing. Section “Research methodology” highlights some related works. The research methodology is discussed in section “Research methodology”. Section “Proposed classification of load balancing algorithms” proposes taxonomy based classification. The results are evaluated in section “Results and discussion” while section “Discussion on open issues on load balancing in cloud computing” discusses upon open issues in cloud load balancing. Finally section “Conclusion and future work” concludes our work and points out some future directions.
Cloud computing concept goes back to the 1960’s when John McCarthy [1] described computation as “computation may some-day be organized as public utility”. Since then efforts have been made and Amazon helped the emerging of cloud computing development by launching Amazon web services as a utility in 2006 [1]. Before this era, users relied on grid computing which was seen as an infrastructure for providing raw computing power like in compute grids or storage of date like in data grids [2]. A new computing paradigm was found the following years which is cloud computing, in this paradigm the resources are provided as a public utility, in a pool, and the user can lease and release those resources via the Internet by an on- demand fashion [3]. The cloud computing has a service delivery model shown Figure 1 in which consists of three layers: (i) Infrastructure as a Service (IaaS) which has the basic hardware to access the service and a collection of those hardware is what forms the datacenter, (ii) Platform as a Service (PaaS) which has the requires operating systems and database management systems, and (iii) Software as a Service (SaaS) which has the applications that are provided to the user in order to use cloud services [4], [5].

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||

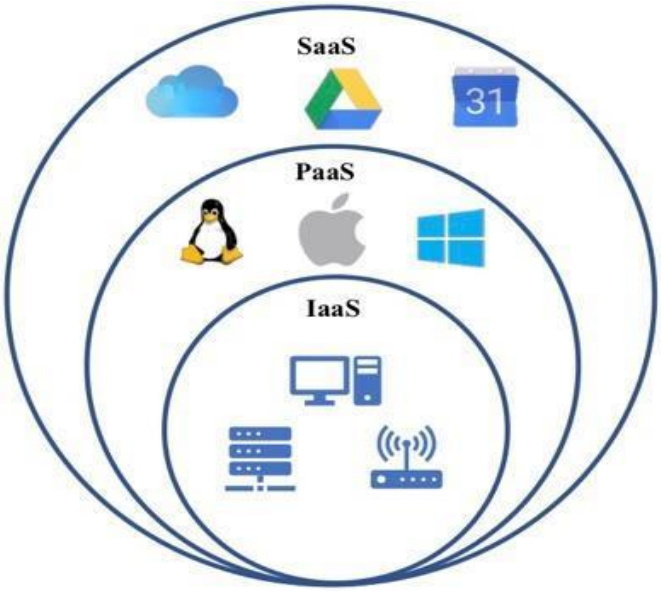


Fig. 1. Cloud Delivery Model

Since the resources in cloud computing is a public utility many users can place a request for those resources at the same time and this calls a need for scheduling to ensure the service it available for every user which is a challenge given the fact that there are many scheduling algorithms to consider and each one of those algorithms has its own factors to keep in calculation whether it is a cost matrix or QoS requirement [1]. Scheduling a task in a system is a tedious procedure and to accomplish it, you need to define few parameters and attributes that are related to the system in which we are scheduling the task or related to tasks we are scheduling. For scheduling in the cloud, we choose the most suitable resources for the execution of a given task in way that provide the minimal time required for the completion of a task [6]. Types of scheduling a task can be divided into two major types static scheduling where it has complete knowledge about the structure of the tasks and the mapping of resources beforehand and it estimates the execution time of the tasks and the other type is dynamic scheduling which does not only depend on the submitted tasks to the cloud environment but also on the system state and the computer machines [6]. In general, an efficient and optimized scheduling algorithm in cloud computing need to consider the following issues like cost, time and service level agreement (SLA) parameters to be followed to as intended by the users [7]. There are also some parameters to consider when choosing a scheduling algorithm like makespan, load balancing, processor utilization, deadline, execution time, completion time, and scalability [6]. In this paper we survey scheduling algorithms used in cloud computing in various settings referencing the applicable scheduling algorithms parameters for each algorithm and at the end providing a rich listing of cloud scheduling algorithms. At the end of the paper an abstract model to facilitate the choice of the suitable algorithms is proposed.

We divide the remaining part of this paper into 5 sections. Findings are reported in Section II, while state of the art results are provided in Section III. The detailed of chooser model are described in Section IV while conclusion and future work is highlighted in Section V.

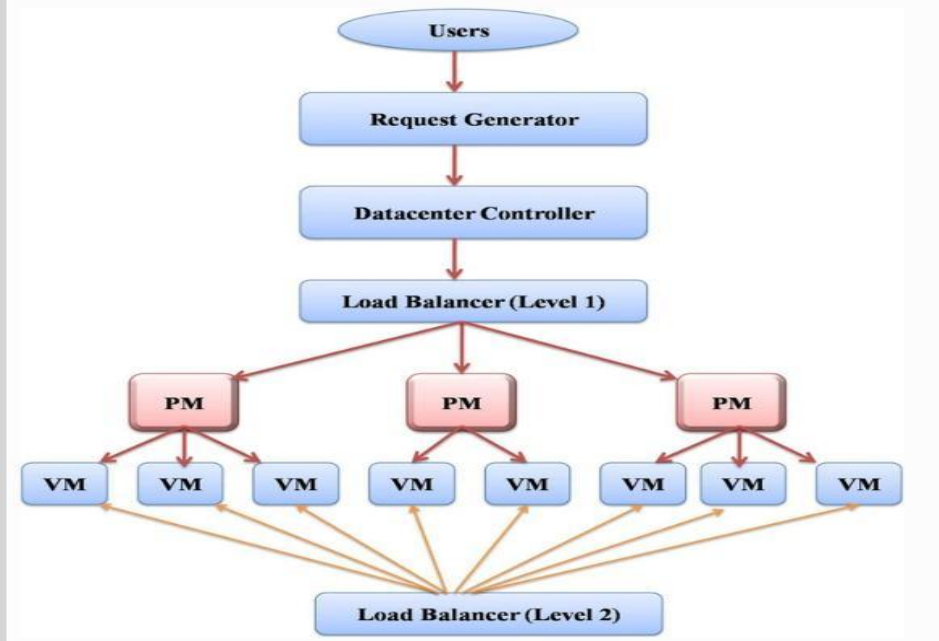
II. FINDINGS

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

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

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||



Two level Load Balancing Architecture

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

Activities involved in load balancing

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

Identification of user task requirements

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

Identification of resource details of a VM

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

Task scheduling

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

Resource allocation

The resources are allocated to scheduled tasks for execution. A resource allocation policy is being employed to accomplish this. A large number of scheduling and allocation policies are proposed in literature. While, scheduling is required for speeding up the execution, allocation policy is used for proper resource management and improving resource performance. The strength of the load balancing algorithm is determined by the efficacy of the scheduling algorithm and the allocation policy [17,18,19].

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||

Migration

Migration is an important phase in load balancing process in cloud and latter is incomplete without the former. Migration is of two kinds in cloud based on entity taken into consideration- VM migration and task migration. VM migration is the movement of a VM from one physical host to another to get rid of the overloading problem and is categorized into types as live VM migration and non live migration. Likewise task migration is the movement of tasks across VMs and is of two types: intra VM task migration and inter VM task migration. A large number of migration approaches has been proposed in literature. An efficient migration technique leads to an efficient load balancing. From the extensive survey it has been concluded that task migration process is more time and cost effective than VM migration and the trend has shifted from VM to task migration [23,24].

In this section, the results of the literature search and selection is shows in two forms which are search execution results and classifies results.

A. Search execution results

The total number of papers found were 32 publications. After applying the inclusion, exclusion criteria 29 publications remained. Finally, after applying quality assessment 27 publications remained.

B. Classified results

Here those 27 publications are classified depending on goal and the outcome of those papers. The classification percentages are shown in Table II below. The percentages show how many papers are under that class.

TABLE I. RESULTS CLASSIFICATION BASED ON GOAL AND OUTCOME

Goal and outcome	Percentage
Model, application, product, and/or service	47%
Review and survey	32%
Challenges and risks	21%

III. STATE OF ART

This section of the paper presents related work done in the field of tasks scheduling algorithms in cloud computing. It is divided into two sections: (i) Reviews and Surveys, that mentions some of the reviews and survey done for scheduling algorithms in the cloud including a review those algorithms proposed by different authors, and (ii) Tabular Comparison of Reviewed Algorithms, which discusses those algorithms that researchers have introduced for the cloud scheduling in regard to parameters like makespan, load balancing, CPU utilization, deadline, response time, and allocation cost.

A. Reviews and Surveys

This section includes the reviews and the surveys done in the field of cloud computing which acted as a building stone for improving existing ones.

The authors in [8] a review on scheduling algorithms of virtual machines in cloud computing where in mentioned the importance of improving existing conventional scheduling algorithms like First Come First Served (FCFS) since they are not suitable for the dynamic nature of the cloud in which the pool of resources is allocated dynamically. They have done a narrative listing of the previous work done on improving the traditional methods whether it is a model, framework, or an algorithm [8]. At the end they have extracted some information from that review which was mainly discussing how conventional algorithms is costly when applied as a virtual machine scheduling algorithms in the cloud compared to those improved algorithms with the intelligence component like Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) [8]. Many factors should be considered when choosing a scheduling algorithm and it mainly dependent on the nature

of the task.

In [9] the have considered the factors of allocations cost which is the cost that a task requires when it is scheduled on the cloud like memory cost for example the other factor is the load balancing factor which is balancing the tasks on the running nodes to distribute the load equally. They compared the traditional FCFS algorithm with some of the optimized and enhanced ones like PSO and Activity Based Costing (ABC) in the CloudSim simulator and the conclusion was that some algorithms performed well in regard to the load balancing factor while failed to reduce the allocation cost factor but mostly the traditional method FCFS required more allocation cost than the algorithms it was compared to [9].

The authors in [10] investigated the scheduling algorithms with regard to Quality of Service (QoS) since it is an importance parameter for the user satisfaction, where they did a flow chart explaining each algorithms discussed in the paper to facilitate the extraction of the advantages and disadvantages and hence draw a conclusion about the studies algorithms.

A survey on many task scheduling algorithms is done in [11] in which they did a comprehensive review on most of the existing algorithms by dividing them into: economic, heuristics, and priority categories and then further dividing those categories into subsection to facilitate the classification of those algorithms by the reader. At the end they produced a paper that can act as a reference for a quick overview or a background search of a certain algorithm [11].

Another survey is done by authors in [12] where they focused their study on static algorithms in which they gave a brief definition of each algorithm and then constructed a performance matrix to evaluate and compare those algorithms. They focused their matrix on make span, time complexity, resource allocation, QoS, average waiting time, and average response time [12]. At the end they provided future work opportunities based on the evaluation they have done on the performance matrix [12].

A review on some scheduling algorithms is done in [6] in which the author provided an overview on FCFS, Shortest Job First (SJF), Round- Robin (RR), and Priority scheduling algorithms. At the end the author of [6] specified the parameters and the objectives for each one of those algorithms which can act as a reference for proposing new algorithms with a certain objective or enhancing existing ones to achieve more objectives.

A review on scheduling algorithms based on meta-heuristics, which is the modification of heuristics algorithms to achieve better performance, method was done in [7] in which they divided the meta-heuristics algorithms to the following classes: Bio-based like GA, Swarm intelligence like PSO, Bat algorithms, and Cat Swarm Optimization. This kind of division is helpful as the heuristic algorithms act as the foundation of many successful new introduced algorithms used for the cloud since scheduling a task in the cloud is considered an Np- complete problem and the best way to deal with it is to improve on heuristics hence having this division can facilitate the way to researchers to introduce more algorithms for the cloud environment.

Author summarized the issues found in [7] providing many opportunities for R&D activities one of them is security aware scheduling which is a major issue since cloud computing relay heavily on conduction all its procedures virtually hence a strong algorithm based on heuristics would act as a practical solution for takes requiring security one of the algorithms to consider is a discrete PSO [7].

The author in [5] discusses how necessary it is to use optimized algorithms for cloud computing rather than the traditional algorithms which are not adapted to the dynamic nature of the cloud. For the main part of the paper the author did a narrative listing if the previous work done in the field and compared them using graphs from the reference studies [5].

A scheduling algorithm based on GA called Tournament Selection Genetic Algorithm (TS-GA) in [13] which has the principle of when a good selection is found instead of removing from the population it is instead introduced to the population while another selection process starts. The TS-GA was compared to RR and GA and registered 32% and 60% more utilization and 7% and 8% less execution costs, respectively [13].

Another algorithm based on GA was proposed in [14] called Improved Genetic Algorithm Task Scheduling (IGATS) which introduces the concept of load priority which was compared to GA in terms of response time and execution time and performed better on both parameters.

Credit based scheduling algorithm is proposed in [15] which considers two parameters, task length and user priority. Since scheduling a task based on task length only may cause starvation and scheduling based on user priority only can decrease the utilization of resources. The authors in [15] compared the proposed algorithm with those focusing on either task length or user priority in terms of makespan and found that the proposed algorithm performed better.

In [16] called Master-Service Uniform Multi-Round (MSUMR) which adopt the conventional master-service model along with Uniform Multi-Round (UMR) with the included restriction of checking the network bandwidth. The performance of MSUMR in terms of completion time was compared to UMR, Multi-Installment (MI), eXtended Multi-Installment (XMI), and One-Batch which are conventional single-round algorithms and MSUMR showed better performance[16].

In [17] the authors introduces a new technique called GreenSched for task allocation and providing resources which is described as an intelligent method since it is an energy aware. Having an energy aware technique that uses CPN-based Green Cloud Scheduling (CGCS) and Forward-only CPN-based scheduling (FCS) in cloud is very useful since energy consumption is usually considered to be a growing problem [17]. CGCS helps locating the nodes with the least energy consumption where FCS identifies the best nodes to use for scheduling [17]. This techniques will still provide QoS measures by fulfilling the deadline and budgets parameters which are specified by the user [17]. They performed some experiments on this technique to prove its application in which authors tested the speed of deadline fulfillment in this techniques by using CGCS, FCS, and SPECweb 2009 [17]. The techniques using CGCS and FCS showed better results that those of SPECweb 2009 [17].

Another paper in which the authors examined the deadline fulfillment is [18] where they proposed a solution to solve the problem of scheduling tasks with deadline and cost constrains on both public and private cloud. They discussed hybrid cloud which helps in determining the where to places the available workload whether it is on a public or private cloud [18]. They introduced a model consisting of two components, public cloud scheduling and hybrid cloud scheduling [18]. In the hybrid the algorithms FCFS and Earliest Deadline First (EDF) are used and there are two parameters extracted which assess whether to run this task load on the public cloud or not and they the unfeasibility and the cheapness [18]. In terms of deadline and allocation cost they noticed after performing their experiments that EDF in both feasible and cheapest settings had the highest deadline meeting percentages with increasing the private cloud CPU and lowest cost over application [18].

In [19] the authors introduced a new algorithm based on Ant Colony Optimization (ACO) algorithm called Load Balancing Ant Colony Optimization (LBACO) algorithm. These algorithms will help in dynamically allocating resources for tasks in the cloud [19]. Their experiments compared their proposed algorithm against ACO and FCFS in terms of load balancing and makespan [19]. The results show that LBACO performed better in both parameters [19].

The authors in [20] proposed a new meta-heuristics cost-effective GA which is designed to meet the deadlines while keeping the cost minimal. The algorithm is called Cost Effective Genetic Algorithm (CEGA) [20]. In their experiments they compared their proposed algorithm against IaaS Cloud Partial Critical Path (IC-PCP), Robust-Cost-Time (RCT), Robust-Time-Cost (RTC), and PSO algorithms in terms of makespan and allocation cost [20]. The proposed algorithms perform better than any other algorithm in makespan parameter whereas in allocation cost parameter it performed better than RTC, RCT, and PSO [20].

In this paper [21] the authors discussed the Multi-Population Genetic Algorithm (MPGA) where the load balancing parameter is take into consideration. The reason they chose the MPGA instead of the GA approach is to avoid the convergence prematurely and they found at the end of the paper that this approach is more suitable for a large number of tasks [21]. Their proposed approach was compared against Time and Cost constrains Genetic Algorithm (TCGA) and Simulated Annealing Genetic Algorithm (SAGA) in term of load balancing, allocation cost, and response time [21]. MPGA showed a better performance in both parameters [21].

In [22] the authors proposed a new algorithms based on combining ACO and PSO called Ant Colony Optimization with Particle Swarm (ACOPS). ACOPS uses the previously saved information for the prediction and adaptation of a new task sets [22]. This algorithm was compared against ACO, FCFS, RR, GA, Aimulated Annealing (SA), Predication mode based

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||

Routing Algorithm based on ACO (PRACO), and Ant Colony System Tree Growth (ACS-TG) in terms of load balancing [22]. The experiment result showed that ACOPS better than other algorithms [22].

In [23] the authors proposed algorithm based on PSO with consideration to load balancing parameters. For this adaptation the authors assumed that the tasks are non-preemptive and independent [23]. The proposed algorithm was compared to RR and improved PSO in terms of makespan, response time, and CPU utilization [23]. In all comparison the proposed algorithm showed better results [23].

As scalability, heterogeneity, and complexity have gained importance, various nature-based [23] computing techniques are used to handle increased sophistication and proficiency. Bio-inspired computing is a fast-growing novel paradigm in communication and networking. Many researchers have been working toward using the role of biological systems and some of the various intelligent algorithms are discussed as:-

3.1 Genetic algorithm

Genetic Algorithm [2],[4] is a searching technique which works randomly based on Darwin theory. It uses current and historical data to analyze the future and this technique is used in VM scheduling. GA is based on the biological concept of generating the population. GA is considered a rapidly growing area of Artificial intelligence. According to Darwin's theory, term “Survival of the fittest” is used as the method of scheduling in which the tasks are assigned to resources according to the value of fitness function for each parameter of the task scheduling process. With the use of evaluation technique, this approach is highly scalable and consumes less energy than first fit decreasing and best fit decreasing. It usually gives the best load balancing results and avoids the migration of virtual machines making it more energy efficient. It considers the task completion time and energy consumed as dual objectives to be fulfilled. This has parallelism imposed internally in it and has the best optimizing ability. It will automatically search itself and itself decides the direction of searching. The increase of virtual machine has no impact on the response time conveying that the system has relatively high performance.

3.2 Simulated Annealing

Simulated annealing is inspired from annealing in solids where annealing in solids such as metal or glass means to heat and allow it to cool slowly, in order to remove internal stresses and toughen it.[4] It is a heuristic approach that has been implemented to obtain optimized solutions of various discrete problems. The origin of the algorithm is in statistical mechanism and the fundamental idea is that the costliest item in the bin is swapped with a random item that has the lowest cost. In this approach, random requests are allocated into a bin until a single parameter in the current bin is completely filled. It is based on two constrains, namely soft constraint which allows a solution to be replaced with better solution and in hard constrain, the capacity of the resource should never exceed the size of the bin. The iterative swapping is done until the solution reached a stable state. This method is highly flexible local search method and can successfully be applied to the majority of real life problems. SA algorithm [28] requires relatively a long time to find a global optimum, it has been demonstrated that the SA algorithm can converge to a global optimum by carefully specifying the cooling rate of the temperature. This algorithm starts by generating an initial solution and by initializing the temperature parameter which is purely an assumption [1] which is the major drawback. This algorithm usually stuck with local maxima, unwanted allocations are entrained and the algorithm also depends on the request availability and bin capacity. This approach works well with high temperatures.

3.3 Tabu Search

Tabu search is a "high level" meta-heuristic, [6]greedy approach for solving optimization problems based on the notion of move. The tabu search algorithm is a global optimization algorithm aiming to simulate human intelligence and has a better ability in local optimization. It is designed to guide other methods to escape
the trap of local optimality, and has been applied to solve resource allocation and other optimization problems The overall approach is to avoid entrenchment in cycles by forbidding or penalizing moves which take the solution, in the next iteration, to points in the solution space previously visited. In Tabu search based
heuristic, we use the solution obtained by Best-Fit heuristic as the initial solution, and adopt random move to find neighbors. The termination condition in the heuristic here is the moving times we required. In general we require the moving times should be twice than the number of the candidates in the solution pool, to increase the probabilities of visiting each candidate solutions through random move. Tabu search uses recency (short term memory) and frequency (long term memory) in order to search a solution space efficient by prohibiting the search from remaining in regions found to be locally optimal and forcing the exploration of other regions not yet encountered. It may encounter the optimal solution for a problem in shorter time that the other traditional methods.

3.4 Ant colony optimization

ACO is a meta-heuristic, population based, [8] stochastic and bioinspired algorithm that mimics the behavior of ants designed to solve combinatorial problems. Ants use chemical substances called pheromones to implicitly communicate with other ants. When an ant explores and finds some target such as food, it secretes a pheromone along the route back to the colony. By doing so, other ants will follow the pheromone to travel down the trail and will also secrete a pheromone if they find the food. However, as time goes by, the pheromone gradually evaporates. Based on the ACO algorithm, the optimization problem can be transformed
into the problem of finding the best path on a weighted graph.[9] The ants incrementally build solutions by moving on the graph. ACO can be used for scheduling generally focuses on reducing the number of physical machines. It achieve better global optimal solutions, have strong robustness achieve better global optimal
solutions, have strong robustness and a a parallel algorithm. It has the disadvantage of overhead and falling for local optima.

3.5 Particle swarm optimization

Particle swarm optimization [10] is a highly advanced heuristic bionic intelligent optimization algorithm that mimics the behavior of animal swarm based on swarm intelligence. It is an adaptive searching algorithm based on group PSO algorithm can remember personal best information and global best information through
collaboration between individuals. Initialize a group of random particles in a space, the particle position represents possible solution, each particle advances to a certain speed, particle swarm gradually approaches to the optimal location after repeated advances which are also called iteration, thus the optimal solution
will be got. In each iteration,[4] the particles update themselves according to two extreme values: one is the optimal solution finding by a single particle, namely the individual extremum; the other is the optimal solution finding by whole particle swarm, that is, namely the global extremum. Each particle in the population represents a possible solution of the problem to be optimized. Its objective is to solve the problem by modeling and predicting social behavior of insects. Instead of using random approach, it serves greater number of virtual machines. It provides the best load balance and reduces the throughput and response time. PSO algorithm is easy to trap into a local optimum. The computational time of the PSO algorithm is shorter than that of other existing metaheuristics, the precision of its final solution to the large-sized complex optimization problems is relatively poor [28]. The PSO algorithm is not robust to solve problems with different constraints The merits of parallel distribution, scalability, easy to realize, strong Robustness, with high flexibility in dynamic environments, PSO solves many combinational optimization problems successfully.

3.6 Artificial Immune System

The AIS is also a general meta-heuristic algorithm based on principles of the immune system, a [7] computational

intelligence approach, called artificial immune system (AIS), has been developed. Artificial immune systems (AIS) constitute a family of bio inspired algorithms based on the models known from the
studies of biological immune systems and immunology . Informally, biological immune systems protect the body from dangerous substances presented in the form of pathogens. They combine the ability to protect from both, general pathogens and specific attackers (e.g. different types of viruses, bacteria and so on) that cannot be be eliminated by the static (innate) part of the immune system . Artificial immune system algorithm includes the negative selection, clonal selection, and immune networks. The clonal selection is based on the selection theory describing the basic response of the immune system to an antigen. Negative
selection is in the biological immune systems used to ensure that newly created lymphocytes will be able to adapt to new types of threats and remain tolerant to body cells at the same time. Immune network contributes to the stable memory structure of the immune system that is able to retain the information about the antigens
even without their presence. It has been successfully applied to many fields such as clustering, classification, pattern recognition, computer defense, and optimization. AIS has the advantage of finding optimal make span values of different size problems.

3.7 Bacterial foraging algorithm

Bacterial foraging optimization algorithm is a global optimization algorithm based on the population [5]search and efficient for global search method for the distributed computing. Bacterial foraging algorithm is a nature inspired optimization techniques that mimicking the foraging behavior of E. coli bacteria satisfies.
It is a multi-criteria optimization problem. It gives optimized solutions to the multi-objective problem where simultaneous multi-criteria are needed to address simultaneously. This method is used for locating, handling, and ingesting the food. During foraging, a bacterium can exhibit two different actions: tumbling or swimming. The tumble action modifies the orientation of the bacterium[16]. During swimming means the chemo taxis step, the bacterium will move in its current direction. It is based on the nature where the selection process tries to preserve those animals that have ability to forage successfully and tries to exclude those animals with poor forage. Since the former have the more ability to succeed in reproduction process. It has the advantage to maximize the resource utilization and min the resource usage cost. It minimizes flow time, make span which are the important scheduling criteria in parallel and distributed computing. [28]

3.8 Fish swarm optimization algorithm

The artificial fish swarm algorithm (AFSA)[11] is a population based meta-heuristic intelligent optimization algorithm inspired from fish swarm behaviors to solve combinatorial problems. AFSA is a random parallel search algorithm belongs to the family of swarm intelligence. This algorithm adapts the behavior of a group of fish swarm intelligence where the group globally searches for the food to reach the areas with a higher
concentration. It then combines this intelligence with one of the methods to get the optimal solution of combinatorial optimization problems. [13]In an AFSA system, each artificial fish (AF) adjusts its behavior according to its current state and its environmental state, making use of the best position encountered by itself and its neighbors. It is highly flexible and more fault tolerant as it can be used for scheduling in cloud. It is expected to give the best results. It is insensitive to initial values, and possesses good performance such as fast convergence and robustness. It has gained an increasing study and wide applications such as multi-objective optimization and clustering problem.

3.9 Cat swarm optimization

An intelligent heuristic scheduling algorithm based on social behavior of cats, belongs to the family of swarm intelligence is cat swarm optimization.[15] It is based on the seeking and tracking behavior of cats. In seek mode, cats sense for the next best move while sitting in a place (without movement) while in tracking
mode the cats chase the target by moving to the next best possible position with a velocity. The cat swarm optimization adapts this behavior of cats with the aim to solve multi objective workflow scheduling. The CSO maps the tasks and virtual machines by taking execution time, data amount and energy consumption as
input and then searches for the best task resource mapping with fairness and offers the best fitness value. The solution obtained optimizes the overall energy consumption. It also provides an optimal task to resource scheduling that minimize the cost of scheduling. It is an improvement over PSO by reducing the number of iterations.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 5, May 2020 ||

3.10 Firefly algorithm

Firefly algorithm (FA) is an intelligent meta-heuristic population based algorithm, inspired by the flashing behavior of fireflies. The firefly method is to dynamically create an optimal schedule based on swarm intelligence, investigating the travel behavior of fireflies which go looking for the closest possible maximum
alternative. The blinking light in the fireflies is their attribute of attractiveness mainly used to attract mates and to defend themselves from other predators. In FA, fireflies are considered as simple agents that move and interact through the search space and record the best solution that they have visited. Therefore, FA can be employed to generate alternative design options in order to effectively support intelligent task scheduling on cloud computing to dynamically map the received jobs to the available resources in order to finish job's execution within minimum makespan time and evenly distribute the load. It can be used to solve multiobjective problems.[17],[18],[19]

3.11 Cuckoo search algorithm

Cuckoo search algorithm is a meta-heuristic algorithm that models natural behavior of cuckoo species. Cuckoos are the beautiful birds but their aggressive reproduction strategy is more interesting to us. The cuckoos reproduce by the following rules where one egg is laid at a time and dumps it in a nest which is chosen randomly and in next step the nest which has the better quality eggs will be carried further for the next generation. It is assumed that there are fixed number of host nests. This strategy can be used for scheduling in cloud where the eggs represent the solution and algorithm works by replacing weaker solutions with the better solutions. This algorithm gives the optimal solution and effectively balances the local and global search with the help of switching parameter. The results obtained are better than particle swarm optimization.[17],[20],[21]

3.12 Artificial bee colony

Artificial bee colony algorithm is a population-based optimization meta-heuristic algorithm based on intelligent behavior of bees, which has two common types; foraging behavior and reproduction (mating) behavior. The ABC algorithm is based on the skillful foraging behavior of honey bee swarm. A typical hive may include 5,000 to 20,000 individual bees. Honey bees assume to have different functions within their colony through time. Active foraging bees go to a food source, check neighbor sources, collect food and return back to the hive. Scout bees examine the area surrounding the hive searching for plentiful new food sources. At any time some of the foraging bees become inactive. This strategy can be applied in the field of task scheduling is the foraging behavior. Mapping of bees foraging behavior to the task scheduling problem, can commonly be defined in a way that the employed bees are associated with allocating tasks on a resource
and share their information about food sources. It minimize the completion time the search through solution space and diversifying the search. These characteristics make the artificial bee colony technique efficient to be applied in cloud task scheduling domain.[21],[12]

B. Tabular View of Reviewed Algorithms

The makespan, load balancing, CPU utilization, deadline, response time, and allocation cost of the algorithms and techniques reviewed above from [13]–[23] are highlighted in Table III.

Algorithm	Makespan	Load Balancing	CPU Utilization	Deadline	Response Time	Allocation Cost
TS-GA [13]		X	High	X	Low	Low
IGATS [14]	X	X	X	X	Low	X
Credit Based [15]	Low	X	X	X	X	X
MSUMR [16]	X	X	X	X	Low	X
GreenSched [17]	X	X	X	High	X	X
EDF (unfeasible and cheapest) [18]	X	X	X	High	X	Low
LBACO [19]	Low	High	X	X	X	X
CEGA [20]	Low	X	X	X	X	Lower than (RTC,
MPGA [21]	X	High	X	X	Low	Low
ACOPS [22]	X	High	X	X	X	X
PSO with Load Balancing [23]	Low	X	High	X	Low	X

TABLE II.TABULAR COMPARISON OF REVIEWED ALGORITHMS AND TECHNIQUES

It can be seen that in each paper the authors compare their proposed algorithms to some of the conventional ones and each author chose different parameters and criteria to consider when evaluating the proposed algorithm. This make the job of choosing the needed algorithm to achieve a task difficult since there are no common grounds which they can follow to compare the available algorithms. To compare two algorithms, you have to know the algorithm steps clearly, which some authors do not state in their papers and you also must execute it in a simulator where you create an environment to run and compare algorithms this can take a substantial time and require a thorough knowledge of the simulator tool. Hence an abstract model, called chooser model, to help in choosing the appropriate algorithm is proposed in section 5 of this paper where the simulation is done in a server and then the appropriate algorithm can be chosen based on that.

IV. WORKING OF CHOOSER MODEL

Based on literature review done in this paper it is clear that the choice of the scheduling algorithm is a difficult yet an important one. Hence an abstract model shown in Figure 2 is proposed to help in solving this problem. This model is appropriate to be used in companies that execute many of their transactions on the cloud since it can be customized to fit the needs of the users.

A. Model's Components

In this section the components of the model in Figure 2 which are User, Check Server, Chooser Server, and Cloud are explained. The algorithms in Figure 2 and Figure 4 use some notations and they are described as follows:

- U is a set of set users parameters received from the checker server

- S is a set of task sets received from the checker server
- x is one task set from the S set of task sets
- t is one task from x task set
- B in one bundle which is a set of many t
- A is one attribute used to form B
- G is one chosen algorithm

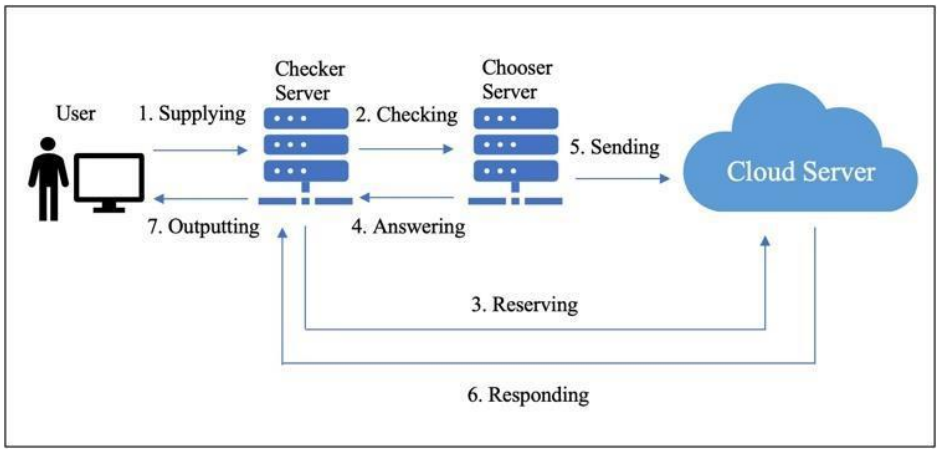


Fig. 2. An Overview of Chooser Model

1) User: The first component in this model is the user who acts as the supplier of the task set to be executed in the cloud. The user is associated with two steps of the model's execution which are, supplying and outputting. This model can support many users simultaneously.

2) Checker Server: The second component of the model is the checker system who is associated with six steps of the model's execution which are, supplying, checking, reserving, answering, responding, and outputting. This server will set a timer and collect many users' entries during that time and then perform its main role. The checker's role is to check two things, the availability of resources in the cloud to execute a task set and checking for appropriate scheduling algorithm by asking the chooser server. When checking the resources in the cloud if the resources are available and the appropriate scheduling algorithms is chosen then those recourses are reserved. This server in its simplest form can be seen as the communication layer between the user and the cloud and can enhance the running time of many existing algorithms since it reserves the resources beforehand. In Figure 3 below is an algorithm showing the working of this server in an abstract fashion.

3) Chooser Server: The third component of the model is the chooser server which is associated with three steps of the model's execution which are checking, answering, and sending. This server does not communicate directly with the user. The chooser server role is to choose the appropriate scheduling algorithm to be used for the task set in the cloud. Many users can use this model hence many tasks sets may arrive at the same time hence another role of this component is to organize the tasks that are independent from different tasks sets by a common attribute and group them in bundles and assign them to the best suited scheduling algorithm based on the supplied parameters by the checker server. The choice of the algorithms is done by applying a Machine Learning (ML) model like Support Vector Machine (SVM) [24] which can compare many algorithms against each other and choose the one with highest match rate in term of meeting the need of the users. Also, this server communicates directly with the cloud in which it sends the task set and the chosen algorithm. In

Figure 3 below is an algorithm showing the working of this server in an abstract fashion.

```
Function Chooser ()
Begin
    Receiving of U and S
    For each x ϵ S
        For each t in x
            If t is independent
                Extract and group in B based on common A
            End if
            Else
                Group in B
            End else
        End for
        Group all B sharing a common A
    End for
    For each B
        For each scheduling algorithm in the server
            Run the algorithm for B using the U
            Choose the algorithm G that meets the U using ML models
        End for
    End for
    Communicate all G with the Checker server
    Send all B to the cloud
End
```

Fig. 4. Pseudocode -code of Chooser Server Algorithm

4) Cloud Server

B. Model's Execution

In this section the execution of the model given in Figure 2 is explained. The execution of the model has many cycles and the order in which those steps are listed does not necessary follow the in sequence.

1) Supplying: The supplying step is where the user supplies the task set to the check server and with that task set the user can specify other parameters like QoS, cost, response time, etc.

2) Chekcing: In this step the checker server checks the chooser server to see if the appropriate scheduling algorithm exists. In this step the checker server supplies the parameters from the user to the chooser server along with the task set.

3) Reserving: In this step the checker server checks the availability of the needed resources to be supplied to the task set execution after choosing the appropriate scheduling algorithm, if there exist the necessary recourses for the task set then those recourses are reserved. This can be done by placing the task set in the queue of those resources. The address of the checker server is saved in this step for the return of the output.

4) Answering: In this step the chooser server replays to the checker server if the appropriate algorithm is found after that the checker server can reserve the appropriate resources in the cloud for execution.

5) Sending: This step is where the chooser server sends the task sets and the chosen algorithms to be executed in the cloud within the reserved resources which were reserved in the reserving step.

6) Responding: This step is executed after the task sets finish executing in the cloud then the results will be sent back to the checker server using the address saved in the reserving step.

7) Outputting: The checker server in this step simply sends the output from the cloud to the user.

V. CONCLUSION AND FUTURE WORK

Traditional approaches of task scheduling suffer from disadvantages like overpricing and slow processing for bulk of tasks. These scheduling algorithms are mostly based on cost reduction factor, deadline factor and even on priority based

scheduling and they suffer from long waiting priority queues. The intelligent scheduling algorithm provides a fair solution to all of these challenges and provides a highly optimized, dynamic and
more reliable scheduling scheme than the traditional. This paper overviews intelligent scheduling algorithm with metaheuristic, swarm based approach. GA, SA and TS have less execution time. ACO reduces the number of physical machines required while PSO, CS give fair distribution. AI, FS, BF, ABC works well with less makespan. CS provides optimal resource utilization. FF, CS, BF increases convergence speed. CS, PSO has relatively less cost.

#	Algorithm	Parameters	Major Features & Disadvantages
1	Genetic Algorithm based On Darwin theory [2],[4]	Execution time, Resource utilization, No. of VM, makespan	• Self managing scheme • Global Search solution • Delivery of accurate results • Better resource utilization • Independent of no. of VM • Min. makespan and flowtime
2	Simulated annealing based on Annealing in solids [4]	Execution time Temperature Request availability Bin capacity	• Depends on resource availability and bin capacity • Less Average execution time GA • Stuck with local maxima • Unwanted allocations
3	Tabu Search based on notion of movement[6]	Execution time	• Less execution time • Do not stuck in local optima • Explore new regions • Premature termination
4	Ant Colony Optimization (ACO)[9]	Randomization No. of VM	• Reducing the number of physical machines • Global optimal solutions • Robustness • Falling for local optima
5	Particle Swarm Optimization [24]	Randomization Convergence Cost, Makespan	• Quick Converge local optima • Fair distribution • Minimize makespan • Lacking of reliability
6	Artificial Immune System [7]	Makespan	• Optimal makespan
7	Bacterial Foraging Algorithm [5] [16]	Makespan flowtime	• Reduced makespan • Minimim flowtime
8	Fish Swarm Optimization Algorithm [11],[13]	Makespan	• High efficiency • Less makespan
9	Cat Swarm Optimization Algorithm [15]	Cost Resource utilization No. of iterations	• Reduced no. of iterations • Optimal resource utilization • Fair distribution
10	Firefly Algorithm [17], [23], [18], [19]	Randomization parameter Light	• Automatic subdivision of entire population. • Efficiently deal with multimodality. • Increased convergence speed.
	Cuckoo Search Algorithm	Population size Switc hing	• Global convergence due to switching • Probability factor. • Use of levy flights result in efficient exploration of search

12	Artificial Bee Colony [21], [12]	Makespan Resource utilization	• Avg makespan is less than PSO and ACO • Better exploration of search space • Better Resource utilization

REFERENCES

[1] K. I. Suthakar and M. K. K. Devi, "Resource Scheduling for Big Data on Cloud," pp. 185–205.

[2] F. Oesterle, S. Ostermann, R. Prodan, and G. J. Mayr, "Experiences with distributed computing for meteorological applications: Grid computing and cloud computing," Geosci. Model Dev., vol. 8, no. 7, pp. 2067–2078,2015.

[3] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," J. Internet Serv. Appl., vol. 1, no. 1, pp. 7–18,2010.

[4] K. M. Uma Maheswari and S. Govindarajan, "A survey of various cloud scheduling algorithms," J. Adv. Res. Dyn. Control Syst., vol. 9, no. 7, pp.1–8, 2017.

[5] G. T. Hicham, "Optimization of Task Scheduling Algorithms for Cloud Computing : A Review Optimization of Task Scheduling Algorithms for Cloud Computing : A Review," no. October, pp. 664–672, 2017.

[6] E. Kumari, "a Review on Task Scheduling Algorithms in Cloud Computing," vol. 4, no. 2, pp. 433–439,2015.

[7] P. Singh, M. Dutta, and N. Aggarwal, "A review of task scheduling based on meta-heuristics approach in cloud computing," Knowl. Inf. Syst., vol.52, no. 1, pp. 1–51, 2017.

[8] R. Eswaraprasad and L. Raja, "A review of virtual machine (VM) resource scheduling algorithms in cloud computing environment," Journal of Statistics and Management Systems, vol. 20, no. 4. pp. 703–711, 2017.

[9] S. C. Satapathy, J. K. Mandal, S. K. Udgata, and V. Bhateja, "Information Systems Design and Intelligent Applications: Proceedings of Third International Conference INDIA 2016, Volume 1," Adv. Intell. Syst. Comput., vol. 433, pp. 619–627, 2016.

[10] S. Potluri and K. S. Rao, "Quality of service based task scheduling algorithms in cloud computing," Int. J. Electr. Comput. Eng., vol. 7, no.2, pp. 1088–1095, 2017.

[11] V. K. Reddy, "Articles a Survey of Various Task Scheduling," vol. 1, no.1, pp. 1–8, 2013.

[12] M. Sohani, S. Jain, I. Narang, K. Agarwal, and S. Arora, "A Survey of Different Task Scheduling Algorithm in Cloud Computing," vol. 6, no. 4, pp. 1–7, 2017.

[13] J. Garg and G. Bhathal, "Research Paper on Genetic Based Workflow Scheduling Algorithm in Cloud Computing," vol. 8, no. 5, 2017.

[14] J. Ma, W. Li, T. Fu, L. Yan, and G. Hu, "A Novel Dynamic Task Scheduling Algorithm Based on Improved Genetic Algorithm in Cloud Computing," vol. 348, pp. 829–835, 2016.

[15] A. Thomas, G. Krishnalal, and V. P. Jagathy Raj, "Credit based scheduling algorithm in cloud computing environment," Procedia Comput. Sci., vol. 46, no. Icict 2014, pp. 913–920, 2015.

[16] T. Zhao and M. Jing, "Bandwidth-aware multi round task scheduling algorithm for cloud computing," J. Intell. Fuzzy Syst., vol. 31, no. 2, pp.1053–1063, 2016.

[17] T. Kaur and I. Chana, "GreenSched: An intelligent energy aware scheduling for deadline-and-budget constrained cloud tasks," Simul. Model. Pract. Theory, vol. 82, pp. 55–83, 2018.

[18] R. Van Den Bossche, K. Vanmechelen, and J. Broeckhove, "Online cost- efficient scheduling of deadline-constrained workloads on hybrid clouds," Futur. Gener. Comput. Syst., vol. 29, no. 4, pp. 973–985, 2013.

[19] K. Li, G. Xu, G. Zhao, Y. Dong, and D. Wang, "Cloud task scheduling based on load balancing ant colony optimization," Proc. - 2011 6th Annu. ChinaGrid Conf. ChinaGrid 2011, pp. 3–9, 2011.

[20] J. Meena, M. Kumar, and M. Vardhan, "Cost Effective Genetic Algorithm for Workflow Scheduling in Cloud Under Deadline Constraint," IEEE Access, vol. 4, pp. 5065–5082, 2016.

[21] B. Wang and J. Li, "Load balancing task scheduling based on Multi- Population Genetic Algorithm in cloud computing," Chinese Control Conf. CCC, vol. 2016–Augus, pp. 5261–5266, 2016.

[22] K. M. Cho, P. W. Tsai, C. W. Tsai, and C. S. Yang, "A hybrid meta- heuristic algorithm for VM scheduling with load balancing in cloud computing," Neural Comput. Appl., vol. 26, no. 6, pp. 1297–1309, 2015.

[23] F. Ebadifard and S. M. Babamir, "A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment," Concurr. Comput., no. October 2017, pp. 1–16, 2017.

[24] Rakotomamonjy A., "Variable Selection Using SVM-based Criteria," J.Mach. Learn. Res., vol. 3, pp. 1357–1370,2003.