International Journal of Research in Advent Technology, Vol.9, No.1, January 2021
E-ISSN: 2321-9637
Available online at www.ijrat.org

Load balancing-based Optimization Techniques in Cloud Computing: A Review

Mr. Rupesh Mahajan, Dr. Purushottam R. Patil, Dr. Amol Potgantwar, Dr.P.R. Bhaladhare

Abstract— Cloud computing relies heavily on load balancing, which ensures that all of the resources, such as servers, network interfaces, hard drives (storage), and virtual machines (VMs), stored on physical servers, are working at full capacity at all times. A typical problem in the cloud is load balancing, which makes it difficult to keep the performance of the applications in line with the Quality of Service (QoS) measurement and the Service Level Agreement (SLA) contract that cloud providers are obligated to give to organizations. It's difficult for cloud providers to fairly divide the work between their servers. Multi-objective optimization (MOO) algorithms, ant colony optimization (ACO) algorithms, honey bee (HB) algorithms, and evolutionary algorithms are all examples of this type of method. The foraging activity of insects like ants and bees served as inspiration for the ACO and HB algorithms. The single-objective optimization problems can be solved by these two techniques, though. ACO and HB need revisions to work with MOPs. This paper summarizes the surveyed optimization methods and describes the modifications made to three specific algorithms.

Index Terms— Load balancing, cloud computing, network, virtual machines, ACO, PSO

I. INTRODUCTION

1.1. Cloud Computing.

The term "cloud computing" refers to a type of Internet-based supercomputing infrastructure. It allows for pooling of assets and on-demand distribution of the cluster's hardware and software. The idea behind cloud computing is to use network technology to create a centralized data center out of a collection of previously dispersed computers. By working together, these clusters increase the system's overall computational power. The principle behind cloud computing is that in a decentralized system, each user has quick and easy access to the services they require. There is now a critical challenge in cloud computing [2] of how to increase the average service response rate of the system. Figure 1 displays the outcomes of the cloud computing system. The SPI model classifies cloud computing into these three broad categories: software as a service, platform as a service, and infrastructure as a service. Platform services can't exist without the IaaS that supports them [3]. When it comes to providing SaaS, the platform as a service relies on the underlying infrastructure to do its work [4]. In the same way that web hosting is related to the platform as a service, the two industries share a lot of similarities. The three distinguishing features of cloud computing are: (1) Virtualization is at the heart of cloud computing, allowing for speedy service provisioning and resource allocation. (2) Users access cloud computing over the Internet and make use of services designed with large data sets in mind. (3) Cloud computing's resources can be dynamically increased and customized to meet the needs of users, and consumers are charged for their real usage. Users are relieved of the responsibility of managing them, which lessens their processing load and their reliance on IT competence.

Load distribution across available resources is necessary to achieve high performance and optimal resource usage when dealing with the diverse cloud resources that are dispersed throughout the globe in multiple data centers. Every available node in the cloud takes on an equal share of the cloud's variable workload. In this case, load balancing is employed. The load might be either the amount of data being transferred via a network or the amount of data being used in other places, such as memory or storage. The goal is to optimize the efficiency with which tasks are executed using cloud resources by distributing the load across the available machines as fairly as possible.

Manuscript revised on January 20, 2021 and published on February 10, 2021
Mr. Rupesh Mahajan, School of Computer sciences and Engineering, Sandip university, Nashik, India
Dr. Purushottam R. Patil, Associate Professor, School of Computer sciences and Engineering, Sandip university, Nashik, India
Dr. Amol Potgantwar, Professor, Sandip Institute of Technology and Research Center, Nashik, India
Dr.P.R. Bhaladhare Professor, School of Computer sciences and Engineering, Sandip university, Nashik, India

doi: 10.32622/ijrat.91202107

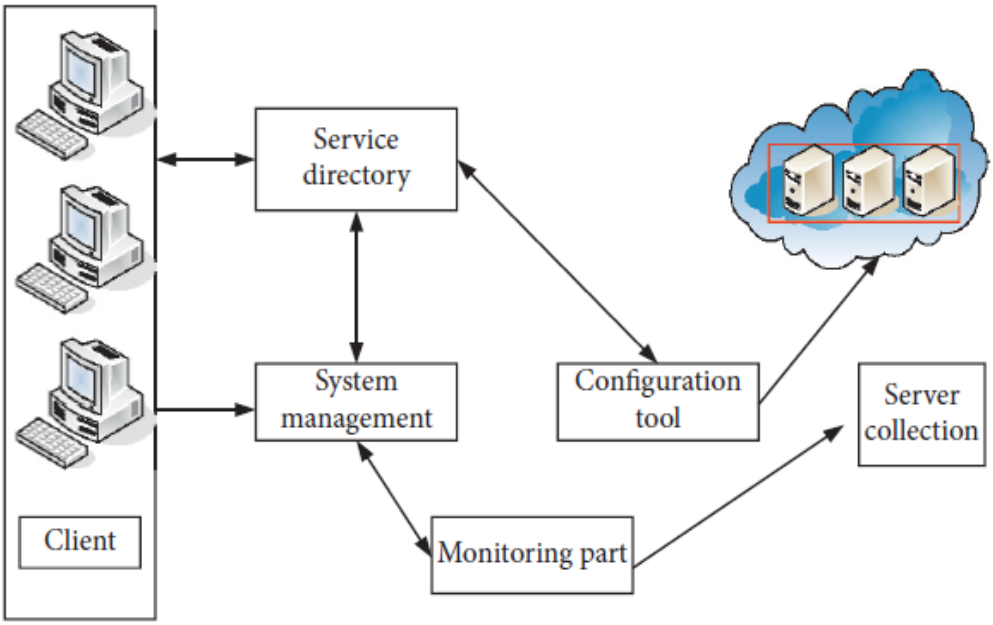

Figure 1: Cloud computing system

1.2 Requirement of Load Balancing Computing resources such as a computer cluster, processing units, computers, central, disc drives, and network links can all benefit from load balancing strategies implemented through a computer network. Strategies for distributing demand that maximize throughput while minimizing reaction time and the risk of asset overload. Reliability can be increased without increasing the workload on any one component by employing load equalization methods. Since it is difficult to predict the number of requests that may be issued to a server, load balancing in the cloud differs from conventional thinking on the load-balancing implementation and design by leveraging product servers to achieve the load balancing. This opens up new possibilities and economies of scale but also brings its own set of difficulties. Cloud computing has many important features, but load balancing is one of the most important [5]. It's a system that evenly distributes the local jobs across all of the cloud nodes such that no one set of machines is ever significantly busier than the others. As a result of its help, we were able to increase both the quality of our presentations and the efficiency with which we used our resources. It also guarantees an expert and equitable distribution of all available computing resources [6]. Moreover, it prevents system bottlenecks caused by uneven loads. Fair-over, or the de provisioning and in-provisioning of cases of applications without fail, is a feature of load balancing that allows services to continue even if one or more of their components fail. illustrates the need for Load Balancing in the cloud when serving numerous users' requests. Many factors, including response time, performance, throughput, scalability, fault tolerance, associated overhead, migration, usage, and time resources, are taken into account by today's load balancing systems in the clouds. With the explosion of data, internet-connected devices, and user requirements, a new paradigm of cloud computing is being developed to deal with these issues. That raises the question of whether or not our cloud model can keep up with the growing demands of an efficient deployment strategy. [7]

II. CHALLENGES OF LOAD BALANCING

Overhead Associated- By employing a load-balancing mechanism, we can quantify the overhead involved. Overhead from completing various tasks, including those that need communication, have been compiled. The effectiveness of a load-balancing algorithm depends on minimizing its overhead.

Throughput - By definition, throughput is the rate at which work may be completed within a certain time limit. The system's efficiency increases with a high throughput. Performance - The term "performance" refers to the system's effectiveness. It needs to be enhanced.

Resource Utilization - The usage of resources can be evaluated using this tool. For a well-balanced system, it needs to be at its maximum. Scalability - Quality of service must remain constant regardless of the total number of users, a feature known as "scalability." The more nodes that can be added without compromising service, the better.

doi: 10.32622/ijrat.91202107

Response Time - In a highly distributed system, the response time is the length of time it takes for one node to react to a request from another node, as determined by the load balancing algorithmic rule. This parameter needs to be shortened for good presentation.

Fault Tolerance – When a node fails, yet the system can still evenly distribute the load is known as fault tolerance. The most reliable method of dealing with failure is load balancing.

Single Point of Fault - The system was built so that the loss of a single component would not disrupt service delivery. A load balancing system must be designed to account for the possibility of a catastrophic failure of a single node, as in a federal system, and to allow for the system as a whole to continue functioning normally.

Period of Migration - Duration The amount of processing time needed to move a single job from one computer to another should ideally be small.

III. LOAD BALANCING ALGORITHMS

Below, we describe the load balancing algorithms currently used in the cloud, along with some important caveats:

A. Random: Static in nature, the random load balancing method is typically defined during system design or development. A random number generator is used to select the node.

B. FCFS: First Come First Serve (FCFS) is a simple load balancing method in which each load balancer maintains a work queue in which jobs wait to be executed. Because of its speed and ease of use, FCFS has many benefits. However, in the event that smaller activities have to wait for longer period because they are further down the queue, the overall response time will suffer.

C. Round Robin: The Round Robin algorithm assigns nodes to fulfil requests in a round robin fashion for a specific time slice, i.e. according to the locally preserved method distribution order. In a scenario when the approaches are equally burdened, this expedites responses. However, different approaches have different work processing times. As a result, certain nodes may be heavily taxed while others sit idle.

D. Weighted RR: Different machines are given different ration weights in technique D, weighted RR. This algorithm ensures that the define ratio weight is proportional to the average number of networks accepted by each machine throughout time. We can provide weighted assignments such as "Machine 1 is able to serve 3x the load that machines 2 and 3 are able to bear," and the load balancer will send three requests to Machine 1 for every request to the others, making this approach superior to Round Robin. While this method operates well, it has a flaw due to the initial static definition of the weights.

E. Dynamic Round Robin: This algorithm is very similar to Weighted Round Robin, with the exception that the servers are always being monitored, and the weights are constantly being adjusted. This is a form of dynamic load balancing. The current number of connections at each node and the fastest node response time are just two examples of how real-time server presentation analysis is used to distribute connections. The sole drawback to this approach is that its dynamic nature makes it difficult to implement in a static load balancer.

F. Equally Spread Current Execution Load: In order to queue up jobs and point them ended to various virtual machines, the F. Equally Spread Current Execution Load method requires constant monitoring of jobs which are present for execution. Based on the scale of the workload, virtual machines (VMs) are assigned to perform it based on how quickly and easily they can complete the task at hand, with the maximum possible output for the least amount of time spent.

G. Throttled Load Balancing Algorithm: When a client request is received, the load balancer tries to find an appropriate VM to complete the required task. The first step of the algorithmic method is to keep a catalogue of all the available virtual machines. Collection is carried out to hasten the process of operation. If a client's request is compatible with the machine's specifications and availability, the request is granted. Next, the virtual machine (VM) is given to the customer. If no available VMs meet the requirements, the load balancer does nothing with the request.

H. Min-Min Algorithm: The first step of the H. Min-Min Algorithm is to find the shortest time required to finish all activities. Next, the minimum value of all the tasks on any supply is selected, and the tasks are scheduled on the corresponding machine. The new total execution time for the machine is then calculated by adding the time it took to complete the assigned work to the total execution time of all other tasks running on the machine. Once a job has been assigned to a machine, it is removed from the pool of available jobs. This process is repeated until the resources have been assigned to all of the tasks. hunger is the main drawback of this method.

I. Max-Min Algorithm: The first technique we'll look at is the Max-Min algorithm, which works similarly to the Min-Min algorithm except that instead of starting with the minimum value, it immediately moves on to designating the maximum value. According to the comparable machine, this is the longest possible period of time that any single task can take up. After that, the time it takes to complete the assigned work is added to the total time it takes to complete all tasks on that system. After a job has been assigned to a machine, it is removed from the list of open jobs. This method is repeated until all jobs have been assigned to available resources.

J. Token Routing: The method was developed with the intention of reducing the overall system cost by redistributing tokens. Agents do not have access to sufficient information for dividing up work because of a backlog in communication. Token-based load balancing, a heuristic approach, mitigates this algorithm's flaws and improves its speed and efficiency

doi: 10.32622/ijrat.91202107

in making routing decisions. Managers here can make decisions about where to pass the token using their own internal knowledge base without needing complete information about the global status or the task loads of their neighbours. There is no added communication cost because the information is simply extrapolated from previously received tokens. j. Genetic Algorithm: Genetic Algorithm is guided by the placement rule, the distribution rule, and the selection rule. As a result of the server scheduling, the job will be completed using a dynamic technique. A number of tasks will be assigned at the outset. After then, the job size will be randomly chosen and executed automatically. The random number generators are deposited in a task slot and the tasks are then processed. Although the evolution of cloud computing is typically unpredictable, it may be defined in terms of the allocation of N jobs submitted by cloud clients to M processing units in the cloud. Each CPU has what's called a "processing unit vector," or PUV, which is a vector of millions of instructions per millisecond (MIPMS). The delay cost, R, and the instruction execution cost, x, are represented graphically.

K. Ant Colony Optimization: Solitary ants exhibit significantly more primitive behaviour than other kinds of bugs. Because of their short-term memory and seemingly random actions, we can only assume that they are not particularly intelligent. By cooperating as a group, ants are able to complete a variety of challenging tasks with remarkable reliability and precision. While this is essentially self-organization prior to learning, ants must cope with a marvel that looks severely like overtraining in enhancing learning strategies algorithmic approach based on ant behaviour. The ACO is the most innovative and well-respected field in the endeavour to create algorithms inspired by ant behaviour, specifically the capacity to identify what computer scientists would term shortest pathways.

L. Particle Swarm Optimization: Particle Swarm Optimization (PSO)-based Task-based System Load Balancing (TBSLBPSO) achieves load balancing in a system by selectively migrating new jobs from a heavily utilised virtual machine (VM) rather than migrating the entire VM. The innovative host VMs were selected, and the additional jobs were migrated using a PSO optimization model. It was found that the TBSLB-PSO procedure dramatically cuts down on the time needed for the load balancing technique compared to conventional load balancing procedures, and that the running of loaded VMs is not interrupted in the process of migration. Additionally, it was found that the VM pre-copy technique is not required. It improved cloud users' Quality of Service by getting rid of VM outages and the possibility of losing their most recent actions.

M. Firefly Algorithm: We can now idealise some of the fireflies' atypical characteristics in order to create algorithms that are inspired by fireflies. For the sake of clarity, we will now define our innovative Firefly Algorithm (FA) using the three simplified principles below: Since (1) all fireflies are unisex, any one firefly will be puzzling to any other firefly regardless of sex, and (2) a firefly's attractiveness depends on its brightness, the less happy firefly will always be attracted to the brighter firefly in a pair of unequally happy fireflies. Both the appeal and the brightness decrease with increasing distance. If there is no more brilliant firefly, it will move at random; (3) a firefly's brightness is showy or determined by the terrain of the neutral function.

IV. LITERATURE SURVEY

Based on xen cloud technology (hypervisor) and the credit system (default scheduler for xen), a compare-and-balance algorithm is suggested in [8] to dynamically assign resources according to demand and distribute workload among servers. The algorithm monitors the CPU and RAM loads on a regular basis, and scales up accordingly. They migrated from a lower threshold value of 10 to a higher range of 70–95 if the necessary resources were unavailable (upper). The suggested GA-based algorithm in [9] uses a genetic algorithm (GA) in conjunction with a gel to solve the load-balancing problem between virtual machines (Vms) (gravitational emulation local search). GA is naturally global in its approach to the search arena in which a gel operates. Using mutation, crossover, and selection, the authors identify two fitness roles. In [10], a compare-and-balance algorithm based on sampling was proposed in order to arrive at an equilibrium solution that reduces the time it takes to migrate virtual machines by using shared storage and satisfies the requirement for zero-downtime relocation of virtual machines by converting them into red hat cluster services. In [11], the authors suggest a PSO-based algorithm for optimising Vm allocation and reaction time. For the implementation, they employ cloudsim technology. It monitors the health of the server and distributes the workload across the available Vms in an efficient manner. An Ant Colony Optimization (ACO) algorithm was proposed in [12] to implement load balancing in a distributed system for the occf (open cloud computing federation), which contains numerous cloud providers. This algorithm improves upon the previous ant colony Algorithm in many ways. Multi-agent genetic Algorithm (maGA) is a hybrid Algorithm combining GA with multi-agent approaches, and it was proposed in [13] as a means of achieving high performance via GA. To evenly distribute workloads across cloud nodes, [14] suggested an Algorithm based on ACO. Rather of each ant updating its own set of results, this strategy has all the ants updating a shared set of results, which is then gradually expanded and improved upon. Algorithm based on bee colony optimization proposed in [15] by modelling honey bee behaviour optimises nectar (throughput) to achieve maximum throughput. An algorithm based on PSO and the endocrine Algorithm, which takes its cues from hormone behaviour in humans, is proposed in [16]. LB is accomplished through the use of a self-organizing strategy between strained Vms. This approach is organised and is founded on Vm-to-Vm communications. Through the use of PSO's increased feed-back method, overworked VMs are able to offload some of their workload to less busy ones. This self-adaptive ant colony optimization activities scheduling Algorithm was proposed in [17]. The ACO algo is made self-adaptive with the help of PSO algo, which also enhances the pheromone update and calculation processes. An ACO-based algorithm called ACO-vmm (ACO based Vm migration) is suggested in

doi: 10.32622/ijrat.91202107

[18]. In this algorithm, a local migration agent independently keeps tabs on resource use and initiates migrations. In order to prevent needless migrations, the Algorithm considers both the current and target system states during the monitoring phase. To locate a nearly optimal mapping between Vms and pcs, this algorithm uses two distinct ant-like traversal algorithms (pms). The fitness value of the PSO-based algorithm suggested in [19] is changed by tweaking the definition of the particle's position and velocity and the rules for updating. This technique accounts for the specifics of complex networks while developing an appropriate model for assigning resources and tasks. The foraging behaviour of honey bees is used as inspiration for an algorithm published in [20] to efficiently distribute work across virtual machines (Vms). In this model, under-utilized Vms serve as the nectar, whereas over-utilized Vms provide the honey. An GA-based algorithm was proposed in [21], and it is based on the double-fitness adaptive Algorithm, which is a genetic Algorithm for balancing the workload across several tasks in a given period of time (jlGA). The technique not only prioritises the order in which jobs are scheduled, but it also seeks to balance the workload between individual nodes. After using a greedy algorithm to seed the population, the author introduces variance to characterise the distribution of heavy workloads across nodes and applies weights to the multi-fitness function. In [22], an ACO-based algorithm for optimising data centre performance and resource usage is suggested. This algorithm can distinguish between overloaded and underloaded servers and distributes workloads accordingly. In [23], a policy approach for scheduling cloud-based tasks was developed; it uses the same load-balancing techniques as ant colony optimization (lbACO). The procedure is meant to restore harmony to the entire setup. The Makespan balancing time was cut in half using a suggested algorithm in [24] that uses ACO. It can be utilised to improve resource utilisation and performance, as well as save time and money during the processing phase. In this approach, tasks and resources are the sole focal points.

V. CONCLUSION

In this work, we focus on optimization issues connected to load balancing that have a significant impact on system performance. Load balancing optimization strategies are studied in depth. In addition, it provides an overview of three popular algorithms found in engineering applications: evolutionary algorithms; ACO; HB; and MOO. The approaches have been consolidated to reveal the MOACO, MOHB, and ant-bee algorithms. The report also discusses the pros and cons of each individual algorithm. Some optimization methods have distinct drawbacks; for instance, the HB algorithm provides the best results only at a set distance (short path), and ACO has issues with speed and convergence. From a theoretical perspective, we can deduce that the three newly developed algorithms based on the combination of optimization approaches are self-adaptive and more practical than the original methodologies.

CONFLICT OF INTEREST

"The authors declare no conflict of interest".

REFERENCES

[1] J. R. Aviles, M. Toril, S. Luna-Ramirez, V. Buenestado, and M. A. Regueira, “Analysis of limitations of mobility load balancing in a live LTE system,” IEEE Wireless Communications Letters, vol. 4, no. 4, pp. 417–420, 2017.

[2] P. Ren, M. A. Kinsy, and N. Zheng, “Fault-aware load-balancing routing for 2D-mesh and torus on-chip network topologies,” IEEE Transactions on Computers, vol. 65, no. 3, pp. 873–887, 2016.

[3] I. Cabria and I. Gondra, “Potential- K- means for load balancing and cost minimization in mobile recycling network,” IEEE Systems Journal, vol. 11, no. 1, pp. 242–249, 2017.

[4] R. Trestian, K. Katrinis, and G. M. Muntean, “OFLoad: an OpenFlow-based dynamic load balancing strategy for datacenter networks,” IEEE Transactions on Network & Service Management, vol. 14, no. 4, p. 1, 2017.

[5] Suguna, S., & Barani, R. (2015). Simulation of dynamic load balancing algorithma bonfring. International Journal of Software Engineering and Soft Computing, July, 5(1), 1-6.

[6] Dimri, S. C. (2015). Various load balancing algorithms in cloud environment. International Journal of Emerging Research in Management &Technology, July, 4(7), 263-266

[7] Haryani, N., & Jagli, D. (2014). Dynamic method for load balancing in cloud computing. IOSR Journal of Computer Engineering, July-August, 16(4), 23-28.

[8] Achar R, Thilagam Ps, Soans N, Vikyath Pv, Rao S, Vijeth Am. Load Balancing in Cloud Based on Live Migration of Virtual Machines. In 2013 Annual IEEE India Conference (Indicon) 2013 Dec 13 (Pp. 1-5). IEEE.

[9] Dam S, Mandal G, Dasgupta K, Dutta P. Genetic Algorithm and Gravitational Emulation Based Hybrid Load Balancing Strategy in Cloud Computing. In Computer, Communication, Control and Information Technology (C3it), 2015 Third International Conference on 2015 Feb 7 (Pp. 1-7). IEEE.

[10] Zhao Y, Huang W. Adaptive Distributed Load Balancing Algorithm Based on Live Migration of Virtual Machines In Cloud. In Inc, Ims and Idc, 2009. Ncm'09. Fifth International Joint Conference On 2009 Aug 25 (Pp. 170-175). IEEE.

[11] Ashwin Ts, Domanal Sg, Guddeti Rm. A Novel Bio-Inspired Load Balancing of Virtual Machines in Cloud Environment. In Cloud Computing in Emerging Markets (Ccem), 2014 IEEE International Conference On 2014 Oct 15 (Pp. 1-4). IEEE.

[12] Zhang Z, Zhang X. A Load Balancing Mechanism Based on Ant Colony And Complex Network Theory In Open Cloud Computing Federation. In Industrial Mechatronics and Automation (Icima), 2010 2nd International Conference On 2010 May 30 (Vol. 2, Pp. 240-243). IEEE

[13] Zhu K, Song H, Liu L, Gao J, Cheng G. Hybrid Genetic Algorithm For Cloud Computing Applications. In Services Computing Conference (Apscc), 2011 IEEE Asia-Pacific 2011 Dec 12 (Pp. 182-187). IEEE.

[14] Nishant K, Sharma P, Krishna V, Gupta C, Singh Kp, Rastogi R. Load Balancing Of Nodes In Cloud Using Ant Colony Optimization. In Computer Modeling And Simulation (Uksim), 2012 Uksim 14th International Conference On 2012 Mar 28 (Pp.3-8). IEEE.

[15] Yao J, He Jh. Load Balancing Strategy Of Cloud Computing Based On Artificial Bee Algorithm. In Computing Technology And Information Management (Iccm), 2012 8th International Conference On 2012 Apr 24 (Vol. 1, Pp. 185-189). IEEE.

[16] Aslanzadeh S, Chaczko Z. Load Balancing Optimization In Cloud Computing: Applying Endocrine-Particle Swarm Optimization. In2015 IEEE International Conference On Electro/Information Technology (Eit) 2015 May 21 (Pp. 165- 169). IEEE.

[17] Sun W, Ji Z, Sun J, Zhang N, Hu Y. Saaco: A Self Adaptive Ant Colony Optimization In Cloud Computing. In ,.Big Data And Cloud Computing (Bdcloud), 2015 IEEE Fifth International Conference On 2015 Aug 26 (Pp. 148-153). IEEE.

[18] Wen Wt, Wang Cd, Wu Ds, Xie Yy. An Aco-Based Scheduling Strategy On Load Balancing In Cloud Computing Environment. In 2015 Ninth International Conference On Frontier Of Computer Science And Technology 2015 Aug 26 (Pp. 364-369). IEEE.

[19] Pan K, Chen J. Load Balancing in Cloud Computing Environment Based On An Improved Particle Swarm Optimization. In Software Engineering and Service Science (Icsess), 2015 6th IEEE International Conference On 2015 Sep 23 (Pp. 595-598). IEEE.

[20] Babu Kr, Joy Aa, Samuel P. Load Balancing Of Tasks In Cloud Computing Environment Based On Bee Colony Algorithm. In2015

doi: 10.32622/ijrat.91202107

Fifth International Conference on Advances In Computing And Communications (Icacc) 2015 Sep 2 (Pp. 89-93). IEEE.
[21] Wang T, Liu Z, Chen Y, Xu Y, Dai X. Load Balancing Task Scheduling Based on Genetic Algorithm in Cloud Computing. In Dependable, Autonomic and Secure Computing (Dasc), 2014 IEEE 12th International Conference On 2014 Aug 24 (Pp. 146-152). IEEE.
[22] Gupta E, Deshpande V. A Technique Based On Ant Colony Optimization For Load Balancing In Cloud Data Center. In Information Technology (Icit), 2014 International Conference On 2014 Dec 22 (Pp. 12-17). IEEE.
[23] Li K, Xu G, Zhao G, Dong Y, Wang D. Cloud Task Scheduling Based On Load Balancing Ant Colony Optimization. In 2011 Sixth Annual China Grid Conference 2011 Aug 22 (Pp. 3-9). IEEE.
[24] Kaur R, Ghumman N. Hybrid Improved Max Min Ant Algorithm for Load Balancing in Cloud. In IEEE International Conference on Communication, Computing & Systems (Icccs–2014).

doi: 10.32622/ijrat.91202107