IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Special Issue-10, May 2019 E-ISSN: 2347-2693

# Task Schduling With Improved ACO In Cloud Computing

**Sushmita Barsainya[1*], Anshul Khurana[2]**

[1,2] CSE, SRIT, RGPV, Jabalpur, India

*Corresponding Author: sushmitabarsainya@gmail.com, Tel.: +91-9406527331*

DOI: https://doi.org/10.26438/ijcse/v7si10.175180 | Available online at: www.ijcseonline.org

***Abstract***— In the current scenario, Cloud computing carved itself as an emerging technology which enables the organization to utilize hardware, software and applications without any upfront cost over the internet. A very efficient computing environment is provided by cloud computing where the customers or several tenants are in need of multiple resources to be provided as a service over the internet. The challenge before the cloud service provider is, how efficiently and effectively the underlying computing resources like virtual machines, network, storage units, and bandwidth etc. should be managed so that no computing device is in under-utilization or over-utilization state in a dynamic environment. A good task scheduling technique is always required for the dynamic allocation of the task to avoid such a situation. The utilization of resources is to be scheduled efficiently so that it helps in reducing the time for task completion. This is task scheduling which is most essential and important part in cloud computing environment. In task scheduling allocation of certain tasks to particular resources at a particular time instance is done. There are different techniques that are proposed to solve the problems of task scheduling. Through this we are going to present the new Algorithm based on task scheduling technique, which will distribute the load effectively among the virtual machine so that the overall response time (QoS) should be minimal. A comparison of this proposed Algorithm of task scheduling technique is performed on workflow simulator which shows that, this will outperform the existing techniques like FCFS, SJF and Genetic Model techniques.

***Keywords***— Cloud Computing, Task Scheduling, FCFS, SJF, Genetic Algorithm, QoS

## I. INTRODUCTION

Efficient task scheduling and resource management is a challenging problem of distributed computing but it is still in its infant stage in spite of exhaustive research in recent years [1][2]. Recently, Amazon developed a novel Elastic cloud based solution in which all the components (PMs, VMs, load balancers etc.) shrink when there is a less load, and expand when there is an increase in the load (here load refers to the incoming tasks for the cloud) [3], [4]. Thus, this elastic cloud resolves the problem of allocating the resources depending on the load. But the efficiency of the VMs depends on the scheduling and load balancing techniques rather than allocation of resources dynamically for its execution. Even though the property of elasticity improves the performance of the cloud, but this service has its own limitations with respect to heterogeneity. For example, scaling of cloud resources with the different system configurations needs to be considered for heterogeneous environment. If the cloud has to work seamlessly, then the load balancer has to play a vital role in satisfying the above mentioned services given by any cloud [5]. There is a significant trade-off between the aforementioned goals in the context of efficient utilization of VMs and managing the resources allocated to each task so that the resources can be utilized efficiently. Scheduling and Load balancing play an important role in allocating the incoming tasks and managing the resources dynamically as demanded by tasks respectively in the cloud environment.

In the present scenario of cloud, efficient utilization of VMs and effective utilization of the resources (CPU and Memory) are very important. These resources must be used in an intelligent manner to attain high throughput and low response time for execution of incoming tasks which are converging from all over the world [6], [7].

Nowadays, many clients demand for several resources (CPU, Memory etc.) on pay-as-you-go model and satisfying these types of tasks is very tedious and needs proficiency in managing these tasks from IaaS platform. Since, the efficiency of the cloud should not be hindered; hence throughput of the cloud should be directly proportional to the efficient utilization of cloud resources. Special focus is needed when these tasks change their demand for resources which are uncertain and handling the resources demand would be a great challenge in the cloud environment. Peer research algorithms such as Round Robin, First Come First Serve, and Throttled [8], [9] etc. have been used for task scheduling and resource management. These algorithms are rule based techniques and suffer from underutilization of

 

VMs, resources; resulting in overall delay in executing the tasks. Further, Bio-Inspired techniques such as Particle Swarm Optimization (PSO) [10], Ant Colony Optimization (ACO) [11], Cat Swarm Optimization (CSO) [12], Honey Bee etc. [13] are much suitable for solving the Scheduling, Resource Allocation and Management problems which come under NP-hard/NP-complete complex classes. Hence, by applying Bio-Inspired techniques, we would enhance the efficiency in terms of reliability, flexibility and time (response and execution). In this paper, we mainly focus on scheduling the tasks using modified ACO algorithms.

Section II contains related work carried out previously. Section III describe proposed method, Section IV represents result and performance evaluation. Section V contains conclusion and future work.

## II. RELATED WORK

A load balancing task scheduling algorithm based on weighted random and feedback mechanisms is proposed by Qian et al. [14]. In the first stage of the algorithm, the chosen cloud scheduling host selects resources by demands and sort them based on static quantification. Secondly, the algorithm randomly chose's resources from sorted list based on their weight; then it obtains corresponding dynamic information to make load filter and sort the remaining. At last, it reaches, in a self-adaptively manner, the right system load through feedback mechanisms. The experimental results show that the algorithm avoids the system bottleneck effectively and achieves balanced load as well as self-adaptability to it.

Malik et al. [15] developed an efficient priority based round robin load balancing technique which prioritizes various tasks to virtual machines on the basis of various metrics, such as required resources or processors, number of users, time to run, job type, user type, software used and cost. Then, it conveys them to various available hosts in round robin fashion. This approach seems to improve the system capability by enhancing various parameters such as fault tolerance, scalability and overhead, and by minimizing resource utilization and response time. This technique has been simulated and tested over CloudSim, which is a widely used tool to test cloud based techniques.

In another research [16], a hybrid ACHBDF (Ant Colony, Honey Bee with Dynamic Feedback) load balancing method for optimum resource utilization in cloud computing is presented. The developed ACHBDF method uses combined strategy of two dynamic scheduling methods with a dynamic time step feedback method. The proposed ACHBDF utilizes the quality of ant colony method and honey bee method in efficient task scheduling. The used feedback strategy checks system load after each phenomenon in a dynamic feedback table to help migrating tasks more efficiently in less time. An experimental analysis comparing existing ant colony optimization, honey bee method and ACHBDF, showed superiority of ACHBDF.

In a Dynamic Resource Scheduling Algorithm, a time delay algorithm is proposed in which the calculation of the arrival time of a request for various tasks is considered as a basic concept. When it comes to scheduling tasks, the job that has more workload (execution time) is initially set. If multiple workloads are the same, they will be randomly selected. That is, if a small job is entered, it will be delayed. Therefore, the delay algorithm analyzes and assigns requests according to the priority and deadline. This algorithm helps to allocate resources to users without delay in processing. [17]

In the other algorithm, the three algorithms RR (Round Robin), SJF (Shortest Job First), and Priority are used, with all processes that is in priority queues. The priority range is low, medium to high. Regarding the RR algorithm, a fair quantum is given to all processes. Therefore, low priority processes are less time-consuming than initial processes, and moderate priority processes have less time than high priority processes. Given the RR algorithm, every job has a time slice for using the processor. Quantum time should not be low because it creates a large number of switches. It should not be too high because it causes starvation. In the SJF algorithm, a process that is less time-consuming in the queue takes the CPU first. Although this algorithm has the best ability, it does cause the processes to go starvation at a longer time. The priority algorithms also place the CPU on top priority tasks, and if the two jobs have the same priority, then use the First Come First Service (FCFS) algorithm.

Regarding the defects of the two popular Min-Min and Max-Min algorithms, it is proposed to overcome these algorithmic errors, first before processing the ETC matrix (Expected Time to Compute) using Min-Max normalization to scale the values in the range Characterized. After normalizing, the new values of the NETC are matrix between 0 and 1, which is most used to select tasks. After normalizing, the last completion time and the first time you finish each task are found on existing virtual machines. The task that takes maximum time difference and planned on the virtual machine. Here, the desirable virtual machine for each task is defined as the virtual machine, which takes the minimum time to perform this task. [18]

In a pre-allocation strategy for planning jobs in cloud computing, a new PACO (Pre-allocation Ant Colony Optimization) algorithm has been introduced based on ACO (Ant Colony Optimization). This algorithm combines an improved ACO algorithm and template size for independent job scheduling. The proposed strategy and algorithm performed well in the simulation environment. These

 

experiments show that PACO can improve scheduling performance. [19]

With the aim of improving the SJF scheduling algorithm, an algorithmic cloud computing algorithm is proposed to obtain the best result of scheduling, lower execution time, and decreasing Makespan. The main modified SJF main idea is to sort out tasks in ascending order based on the length of jobs and calculating the average of the total length of the tasks. Then the algorithm checks for the length of all jobs, if it is less than the average job length, and the number of tasks in VM1 is less than the number of tasks in VM2, then the job is sent to VM1 instead of sent to VM2. In this paper, the SJF job scheduling algorithm is modified to achieve the scheduled tasks with the minimum Makespan. It focuses not only on the completion time of the jobs, but also on the completion of all tasks. We demonstrated with empirical evidence that MSJF is better than SJF and FCFS. Experimental results showed that MSJF (Modified Shortest Job First) was more efficient in improving response time and Makespan compared to SJF and FCFS. [20]

The algorithm was presented with Priority Aware Longest Job First (PA-LJF). In those users, they send things through the web interface. The tasks are with an assistant receiving the request. The user who sent the job is authenticated if the job sent from him is sent to the priority queue, otherwise the user will be known as the attacker and his job will not be performed. After confirming the user, the tasks are arranged by their size and descending order and the virtual machine that can do it faster will take the job. This is done by the resource regulator, which has many scheduling algorithms. Including min-min, PSO, ACO, greedy resource allocation, honey bee, round robin and more. Each of these algorithms has advantages and disadvantages. Therefore, according to the user's need or choice, the scheduling algorithm is selected. After the end of the task, the algorithm's performance is minimized, minimum run-time, high efficiency, response time, bandwidth used and user satisfaction is checked. [21]

In optimization algorithms, optimization can be used to improve scheduling parameters such as cost, energy, resource usage, rotational time, and runtime. An algorithmic resource was proposed to reduce Makespan and increase resource productivity. In this algorithm, it provides scheduling for a large number of independent tasks to examine the performance of the Makespan and CPU parameters based on the particle optimization integration and the Max-Min algorithm. Therefore, this algorithm combines the advantages of both algorithms with good accuracy. The traditional PSO algorithm takes a lot of time to find the optimal solution. But by integrating with the Max-Min algorithm, it outputs for possible sequences of tasks that can be assigned to a virtual machine. This output is then given as input to the Particle Swarm Optimization (PSO) algorithm. As a result, it helps to find the optimal solution sooner than the traditional one. Experimental results show the positive performance of this algorithm with other algorithms. [22]

The RASA (Resource Aware Scheduling Algorithm) algorithm is defined so if the number of resources available is odd, the Min-min strategy is applied to assign the first task, otherwise the Max-min strategy applies. The remaining tasks are assigned to one of two strategies to their respective resources. For example, if the first task assigned to a Min-min strategy, the next task is determined by the strategy of Max min.

In the next round, the assignment of job begins with a different strategy from the last round. For example, if the first round starts with the Max-min strategy, the second round will begin with the Min-min strategy. The time complexity of the RASA algorithm is equal to 2O (MN), where M represents the number of resources and N represents the number of tasks, which is similar to the complexity of the algorithm Max-min and Min-min. [23]

In the Enhanced User Preference-Based Intelligent Scheduling Algorithm (E-UPISA) considers the priority dynamically. For the first time, they provide equality with the entire process and according to the primary priority that users often use. A process that will not be used for a long time will reduce their priority. Experimental results show that E-UPISA reduces the waiting time, rotational time, and response time for the desired processes, and reduces overall system performance compared to conventional programming algorithms. In this paper, user preferences are used as the basis for planning. The proposed algorithm uses dynamic priority criteria. If the user uses some of the processes with respect to time, the priority will increase. If the user does not use an important job for a while, its priority will be reduced. Then the algorithm classifies each process as high, medium, and low priority, and then looks for small things. If tasks are performed with the SJF threshold, the tasks get higher priority, so the waiting time is reduced. Compared to existing algorithms, E-UPISA improves the results. The performance of E-UPISA is not good at the first time, but over time, it learns the user's preference, and this is much better than the usual programming algorithms [24] .

Ashish Gupta, Ritu Garg [25] chooses a meta-heuristic approach of Ant colony optimization algorithm to solve the task scheduling problem in cloud environment focussing mainly on two objectives, i.e., minimizing the makespan/ computation time and better load balancing. Tasks are allocated randomly. High probability has been taken for Data Centre allocation which results in poor allocation of resources.

 

Xiao-Fang Liu et al [26] developed an ACS-based approach to achieve the Virtual machine placement goal. It effectively minimizes the number of active servers used for the assignment of virtual machines (VMs) from a global optimization perspective through a novel strategy for pheromone deposition which guides the artificial ants towards promising solutions that group candidate VMs together. The number of servers provided for placing VMs reduces as the generation number grows, which increases the computation providing guidance for further advancement of the solutions. Learning process is not defined properly for VM allocation to servers.

Yifan Ding et al [27], address the mapping problem as a TSP and propose to apply degradation factor ant colony algorithm technique for optimal placing of virtual machines in the physical servers. The value of degradation factor has been calculated by resource waste model. Even though, the experimental results show that the degradation factor is effective. There is no high efficiency. When leading into deterioration factors, it ignores the complexity of itself algorithm, though it is only a simple pre-treatment, but its computation will also consume a certain amount of computing resources.

## III. PROPOSED METHODOLOGY

To overcome the limitations of existing algorithms we propose a new algorithm that will work with optimal solutions and will always gives the best results by selecting proper virtual machines in the DC of CSP. The proposed algorithm of load balancing has been described below. For independent task scheduling in cloud using ACO technique whose objective is to achieve high load balancing and reduce the makespan. The main concept of ACO is to simulate the searching behaviour of artificial ant's colonies. When group of ants searching for food, they excreta special kind of chemical is also known as pheromone. Firstly, ants randomly start to search their food. When food source is found, they spilt pheromone on the path; ants track the trails of the previous ant's food source by sensing pheromone on the soil. As this procedure continues, the majority of the ants pull towards to choose the best-so-far path as there have been huge amount of pheromones accumulated on this path.

The proposed algorithm is based on ant's searching behaviour. In the exciting method due to random allocation of virtual machine sometimes it may increase the execution time, because large size task allocates to the small configuration/small processing power resource so it can affect the execution time. In the proposed algorithm the task are sorted in ascending order and virtual machine are allocated according to their probability. This reduces the execution time of virtual machines and increases the throughput of the machines.

The estimation of the execution time is denoted by ET (ti,mj) which represents the estimated time for execution of task *i* on machine *j*. It is computed by division of Computational Cost of Task Ci with Computational Cost of Machines Cj. Completion Time

$$CT\ (t_i,m_j) = \text{start time} + ET(t_i,m_j)$$

Proposed algorithm is as follows:

1.Initialization:
Task_List=Φ
Assign incoming tasks to Task_List as per their arrival time.
Task_List ← {t1, t2... tn}
*m* = no. of available VMs
*n* = |Task_List| //total no. of tasks
**Set** Optimal_solution = Φ, α = 1, β =5
**Set** Initial pheromone value = 0.5

**2.Do**
**If** *(n > m)*

3. Evaluate Execution Time based on computing cost of Task and computing capability of VMs.
4. **For** 1 to *Imax* no of iterations
5. Arrange the task to VM (up to No. of VMs) and update the completion time (CT) accordingly.
6. **For** 1 to k ants do
7. **For** tasks ti= 1 to *n* do
8. **For** virtual machines VMj= 1 to *m* do
9. Calculate Heuristic information $\eta$ and Probability transition rule for each VM

$$\eta = 1/d_{ij}$$

Where d is the distance from load balance to virtual machine.

10. Find the probability of each virtual machine using following formula

$$\text{Probability} = \frac{\eta^{\beta} \times \tau^{\alpha}}{\sum_i^m \eta^{\beta} \times \tau^{\alpha}}$$

Where Heuristic information $\eta$

11.Store the probability in increasing order.
12.**End For**
13.Allocate the ∑ task ti to VM VMj based on LOW probability.
14.Update completion time.
15.Apply LB-ACO procedure on input Task_List and m VMs
16.**End For**
17. **Else**
18. Assign all *n* tasks from Task_List to *m* VMs.

## IV. RESULTS AND DISCUSSION

To evaluate result of proposed algorithm and comparing it with existing algorithms Cloud Analyst simulator has been used which requires Netbeans and CloudSim to be installed on system. In this research load balancing policy based on modified ACO algorithm defines data center processing time, data center transfer cost and response time. The attribute

 

used to measure the performance obtained by proposed and existing Cloud service brokering strategies are average response time of cloud users allocated VM to Cloud Service Providers. Cloud analyst is shown in figure 1.

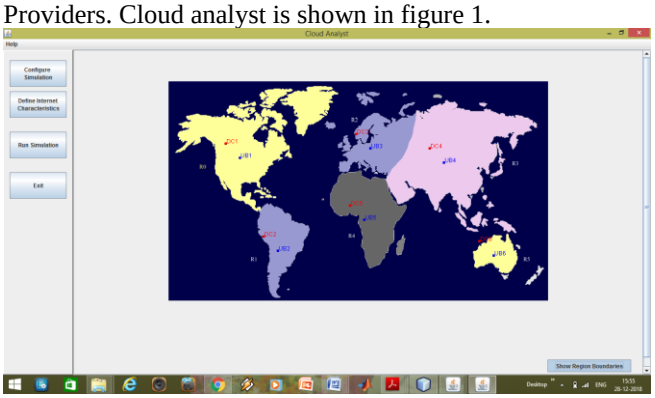

Figure 1: Cloud Analyst

Virtual machines of different sizes are used for this evaluation. Table 1 shows the performance comparison between the proposed algorithm, Existing ACO, Honeybee algorithms and PSO algorithms on the basis of overall response time in milliseconds.

Table 1. Evaluation of Algorithms for overall response time in ms

| No of CU/DC | Algorithms | | | |
|---|---|---|---|---|
| | LBACO | Honeybee | PSO | Proposed |
| 5/3 | 314.56 ms | 326.62 ms | 318.69 ms | 311.06 ms |
| 30/5 | 76.23 ms | 78.65 ms | 78.21 ms | 74.47 ms |

Graphically comparisons are shown below:

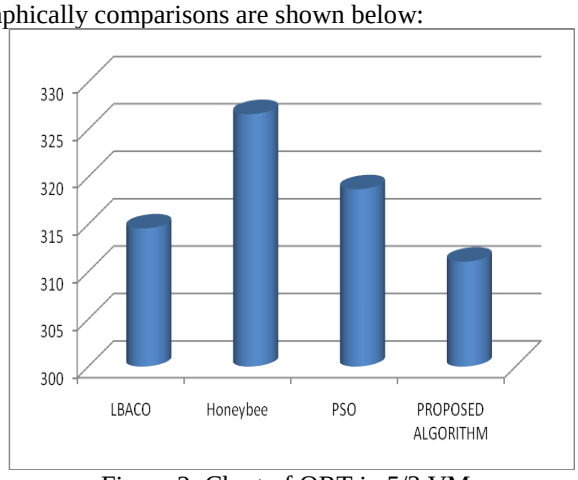


Figure 2: Chart of ORT in 5/3 VMs

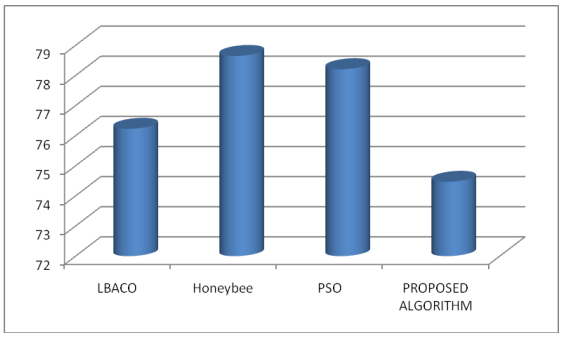


Figure 3: Chart of ORT in 30/5 VMs

## V. CONCLUSION AND FUTURE SCOPE

This thesis analyzes and summarizes currently common loading balance technology and its advantages as well as disadvantages. On this basis, the thesis discusses load balance realization under cloud environment. Based on traditional ACO algorithm, improved aco selection strategy is introduced to set up load balancing model and put forward the improved ACO algorithm-based resources loading balance optimization model. Then, we choose CloudSim extension called Cloud Analyst as final simulation platform to simulate data center, virtual resources and user's task under cloud environment. From user's task completion time and load balancing, we compare the improved ACO algorithm with Round Robin, ACO, PSO and Honeybee algorithm. The experiment result shows our scheme is better and it can be taken as an effective loading balance algorithm under cloud environment. Targeting one of the issues in cloud environment, we have proposed an approach to allocate optimal resource to the incoming requests from the users irrespective of their numbers. Also the load can be cleared using the ant colony optimization technique. Simulation results shows that the proposed approach provide good performance than the compared existing algorithms. As it is NP-Complete to schedule tasks, other new meta-heuristic methods could be found to further improve this vital problem which requires lower run time.

## REFERENCES

[1]. C.-W. Tsai, W.-C. Huang, M.-H. Chiang, M.-C. Chiang, and C.-S.Yang, "A hyper-heuristic scheduling algorithm for cloud," IEEE Transactions on Cloud Computing, vol. 2, no. 2, pp. 236–250, 2014.

[2]. Y. Wang and W. Shi, "Budget-driven scheduling algorithms for batches of map-reduce jobs in heterogeneous clouds," IEEE Transactions on Cloud Computing, vol. 2, no. 3, pp. 306–319, 2014.

[3]. Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," IEEE Transactions on parallel and distributed systems, vol. 24, no. 6, pp.1107–1117, 2013.

[4]. B. Guan, J. Wu, Y. Wang, and S. U. Khan, "Civsched: a communication-aware inter vm scheduling technique for decreased

 

network latency between co-located vms," IEEE Transactions on Cloud Computing, vol. 2, no. 3, pp. 320–332, 2014.
[5]. Q. Zhang, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, "Dynamic heterogeneity-aware resource provisioning in the cloud," IEEE transactions on cloud computing, vol. 2, no. 1, pp. 14–28, 2014.
[6]. A. J. Younge, G. Von Laszewski, L. Wang, S. Lopez-Alarcon, and W. Carithers, "Efficient resource management for cloud computing environments," in Green Computing Conference, 2010 International. IEEE, 2010, pp. 357–364.
[7]. M. Polverini, A. Cianfrani, S. Ren, and A. V. Vasilakos, "Thermal aware scheduling of batch jobs in geographically distributed data centers," IEEE Transactions on cloud computing, vol. 2, no. 1, pp.71–84, 2014.
[8]. K. Al Nuaimi, N. Mohamed, M. Al Nuaimi, and J. Al-Jaroodi, "A survey of load balancing in cloud computing: Challenges and algorithms," in Network Cloud Computing and Applications (NCCA), 2012 Second Symposium on. IEEE, 2012, pp. 137–142.
[9]. B. Radojevic´ and M. Zˇ agar, "Analysis of issues with load balancing algorithms in hosted (cloud) environments," in MIPRO, 2011 Proceedings of the 34th International Convention. IEEE, 2011, pp.416–420.
[10]. X. Zuo, G. Zhang, and W. Tan, "Self-adaptive learning pso-based deadline constrained task scheduling for hybrid IaaS cloud," IEEE Transactions on Automation Science and Engineering, vol. 11, no. 2, pp. 564–573, 2014.
[11]. K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, R. Rastogi et. al., "Load balancing of nodes in cloud using ant colony optimization," in Computer Modelling and Simulation (UK Sim), 2012 UKSim 14th International Conference on. IEEE, 2012, pp. 3–8.
[12]. R. Shojaee, H. R. Faragardi, S. Alaee, and N. Yazdani, "A new cat swarm optimization based algorithm for reliability-oriented task allocation in distributed systems," in Telecommunications (IST), 2012 Sixth International Symposium on. IEEE, 2012, pp. 861–866.
[13]. P. V. Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," Applied Soft Computing, vol. 13, no. 5, pp. 2292–2303, 2013.
[14]. Qian, Z., et al., A load balancing task scheduling algorithm based on feedback mechanism for cloud computing. International Journal of Grid and Distributed Computing, 2016. 9(4): p. 41-52.
[15]. Malik, A. and P. Chandra, Priority based Round Robin Task Scheduling Algorithm for Load Balancing in Cloud Computing. Journal of Network Communications and Emerging Technologies (JNCET) www. jncet. org, 2017. 7(12).
[16]. Pawar, N., U.K. Lilhore, and N. Agrawal, A Hybrid ACHBDF Load Balancing Method for Optimum Resource Utilization In Cloud Computing. 2017.
[17]. Ajay Thomas S. and Santhiya. C, "Dynamic Resource Scheduling using Delay Time Algorithm in Cloud Environment", Second International Conference on Computing and Communications Technologies (ICCCT'17). IEEE, 2017.
[18]. Gupta R., et al., "An Effective Multi-Objective Task Scheduling Algorithm using Min-Max Normalization in Cloud Computing", 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), IEEE, 2016.
[19]. Lin, R. and Q. Li, "Task Scheduling Algorithm Based on Pre-Allocation Strategy in Cloud Computing", IEEE International Conference on Cloud Computing and Big Data Analysis. 2016.
[20]. Alworafi, M.A., et al., "An Improved SJF Scheduling Algorithm in Cloud Computing Environment", International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques, IEEE, 2016.
[21]. Kumar, M. and S.C. Sharma, "Priority Aware Longest Job First (PALJF) Algorithm for utilization of the resource in cloud environment", International Conference on Computing for Sustainable Global Development, IEEE 2016.
[22]. Jain, A. and R. Kumari, "An Efficient Resource Utilization Based Integrated Task Scheduling Algorithm", 4th International Conference on Signal Processing and Integrated Networks (SPIN), IEEE, 2017.
[23]. Parsa, S. and R. Entezari- Maleki, "RASA: A New Task Scheduling Algorithm in Grid Environment". World Applied Sciences Journal 7 (Special Issue of Computer & IT), 2009.
[24]. Kamal, R., et al., "Enhanced User Preference Based Intelligent Scheduling Algorithm (E-UPISA)", Proceedings of the 23rd International Conference on Automation & Computing, University of Huddersfield. 2017.
[25]. Ashish Gupta, Ritu Garg, "Load Balancing Based Task Scheduling with ACO in Cloud Computing", ICCA, IEEE-2017.
[26]. Xiao-Fang Liu, Zhi-Hui Zhan, Jeremiah D. Deng, Yun Li, Member, IEEE, "An Energy Efficient Ant Colony System for Virtual Machine Placement in Cloud Computing"1089-778X (c) 2016 IEEE.
[27]. Yifan Ding, Guang zhong Liao, Siyuan Liu, "Virtual Machine Placement Based on Degradation Factor Ant Colony Algorithm", IEEE-2018.

 