Load Balancing Heuristic for Tasks Scheduling in Cloud Environment

Kadda Beghdad Bey[1], Farid Benhammadi[1], Mohamed El Yazid Boudaren[2] and Salim Khamadja[1]
[1]*Informatics Systems Laboratory, Ecole Militaire Polytechnique, Algiers, 16111, Algeria*
[2]*Ecole Militaire Polytechnique, Algiers, 16111, Algeria*

Keywords: Cloud Computing, Task Scheduling, Resource Allocation, Makespan Optimization, Min-Min Algorithm.

Abstract: Distributed systems, a priori intended for applications by connecting distributed entities, have evolved into supercomputing to run a single application. Currently, Cloud Computing has arisen as a new trend in the world of IT (Information Technology). Cloud computing is an architecture in full development and has become a new computing model for running scientific applications. In this context, resource allocation is one of the most challenging problems. Indeed, assigning optimally the available resources to the needed cloud applications is known to be an NP complete problem. In this paper, we propose a new task scheduling strategy for resource allocation that minimizes the completion time (makespan) in cloud computing environment. To show the interest of the proposed solution, experiments are conducted on a simulated dataset.

1 INTRODUCTION

Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. Cloud computing is a promising paradigm given the increasing demand of high performance computing and data storage. Today, most companies have switched to cloud computing solutions for data hosting and applications running in order to take advantage of its three main services: Software as a Service (SaaS), Infrastructure as a service (IaaS) and Platform as a Service (PaaS). Generally, distributed computing environments aim at running different computational intensive services that have diverse computational requirements by ensuring the scalability, availability and resource utilization. In cloud systems, the same application can be used by multi-clients at the same time while preserving security and private data of each client. This is made possible by using virtualization principle to share a resource between several users.

Resources management in Cloud computing environment is one of the most important challenges. Resource allocation problem is defined as the process of assigning available resources to the needed cloud application over the internet (Vinothina et al., 2012). In literature, several solutions (Jeyaram and Vidhya, 2013; Keshk, 2014) for resource allocation problems in cloud computing system have been proposed. The optimality criterion in resource allocation is to minimize the global completion time (Makespan) while maximizing the resources utilization.

The scheduling problem can be either static or dynamic. Task scheduling algorithms have been designed as NP-complete problems that do not converge to optimal solutions. The scheduling strategy defines the instants when the scheduling algorithm is called to produce a schedule based on forecasting resources performances and independent tasks to be executed. The task scheduler in cloud computing environment is to determine an appropriate assignment of resources to tasks in order to meet up the needs of clients. A large number of jobs scheduling heuristics are available for maximizing profit via resources allocation in cloud computing systems (Kuribayashi, 2011; Abirami and Shalini, 2012; Gouda et al., 2013).

In this paper, we tackle the problem of resource allocation and tasks scheduling problem for cloud computing systems. The Resource allocation problem in cloud computing environments has become a very active research topic. Task scheduling in cloud environment consists in mapping the task onto the available resources and the main objective is to maximize resources utilization and to minimize the total completion time. The task scheduling approach, proposed in this

Bey, K., Benhammadi, F., Boudaren, M. and Khamadja, S.
Load Balancing Heuristic for Tasks Scheduling in Cloud Environment.
DOI: 10.5220/0006240304890495
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 1*, pages 489-495
ISBN: 978-989-758-247-9

paper, relies on a load balanced makespan-based heuristic to achieve a good task mapping on available resources. Jobs are decomposed into various tasks where each task should be assigned to the best suited resources for its execution.

The remainder of the paper is organized as follows. Section 2 presents related works about task scheduling algorithms in cloud environment. The problem of tasks scheduling for resource allocation and the new meta-task scheduling algorithm in cloud computing environments are proposed in Section 3. Simulation results are presented in Section 4; and Section 5 concludes the paper.

2 RELATED WORKS

Cloud computing is one of the emerging technologies in the distributed computing environment to meet the scientific applications demands in terms of computing power and data storage. Cloud computing systems have become an active research topic in recent years. In literature, plethora of heuristic algorithms for resource allocation and scheduling (Goudarzi and Pedram, 2011; Delhi and Giridhar, 2014; Beghdad-Bey et al., 2015) has been proposed. The aim of these scheduling techniques is to determine an optimal allocation of cloud resources, where resources are scheduled based on user requests.

(Goudarzi and Pedram, 2011) presented a new approach based on SLA (Service Level Agreements) for resource allocation problem in cloud along with a distributed solution to resolve this problem. More explicitly, authors present a mathematical model based on Generalized Processor Sharing and a heuristic algorithm in order to maximize the profit for resource allocation. An online scheduling for resource allocation and task scheduling problem in cloud is presented in (Keshk, 2014). The author proposes a load balancing approach using ant colony algorithm to improve the resource allocation. Similarly, (Tawfeek et al., 2015) proposed a new task scheduling algorithm in cloud based on Ant Colony Optimization (ACO). Authors present a comparison between the proposed approach and existing scheduling algorithms as First Come First Served (FCFS) and Round-Robin (RR). They demonstrate through simulation results that ACO algorithm outperforms both algorithms. (Abirami and Shalini, 2012) design a scheduling algorithm named Linear Scheduling for Tasks and Resources (LSTR) in order to improve task scheduling and resource utilization. To establish the IaaS cloud environment, a combination of Nimbus and Cumulus services are imported to a resource node. The virtualization technique used with the scheduling algorithm yields higher resource utilization, and improves the cloud resources performance.

On the other hand, an innovative admission control and scheduling algorithms for efficient resource allocation are introduced by (Irugurala and Chatrapati, 2013) to maximize profit by minimizing cost and improving customer level. (Gouda et al., 2013) use a priority algorithm, based on some optimum threshold decided by the cloud owner, which decides the allocation sequence for different jobs requested among the different users. The same idea can be applied to allocate available resource with minimum wastage and maximum profit. (Silva, 2008) optimize the machines number allocated for processing an analytical task so that maximum speedup can be achieved within a limited budget. Since web applications traffic is dynamic and random, forecasting optimal number of machines allocated to client applications in real-time is not a trivial task.

The PSO (particle swarm optimization) algorithm has also been applied for task scheduling in cloud computing. (Guo, 2013) introduced a PSO algorithm based on small position value rule in order to minimize the cost of the processing. (Gomathi and Karthikeyan, 2013) propose Hybrid Particle Swarm Optimization (HPSO) based scheduling heuristic to balance the load across the entire system while trying to minimize the makespan and maximize the resource utilization. Conventional Min-min and Max-min algorithms are used (Katyal and Mishra, 2014) for task scheduling. Authors apply several heuristics to select the best suited among the two algorithms, in terms of overall makespan, for on-demand resources allocation. In the same context, authors also consider the resource utilization criterion. (Kumar and Verma, 2012) describe a new scheduling algorithm that uses three scheduling algorithms such as Min-Min, Max-Min and Genetic Algorithm. The results obtained show that the improved genetic algorithm can minimizes the completion time. (Fang, 2010) develop a new task scheduling algorithm based on load balancing. This approach can ensure high resource utilization based on simulation results of CloudSim. Moreover, other techniques about task scheduling are detailed (Selvarani and Sadhasivam, 2010; Kanrar, 2012).

3 PROPOSED BALANCED TASK SCHEDULING ALGORITHM

The development and deployment of distributed applications in cloud computing environments is being challenged by the need of scheduling efficiently a large number of tasks and resources. In this work, we present a new scheduling strategy based load balancing (LBE) approach for independent task in Cloud environment. This heuristic aim to minimize the total completion time (Makespan) and improve the utilization of resource.

3.1 Formulation and Modeling Problem

Cloud environment is a promising distributed platform to meet the increasing computational requirements of scientific and technical applications. Cloud computing is a new computing paradigm in which these components consist of physical and virtual resources. The dynamic resources allocation is the most important problem in this area.

Before defining and modeling the task scheduling problem, let us describe the cloud architecture considered in this work. We define the cloud computing architecture as a set of clusters containing a different number and type of computing resources each. Therefore, each resource is constituted of a different number and types of virtual machines (VMs). Figure 1 illustrates the functional architecture of the Cloud computing system which is composed of three main layers:

- Client Layer: represents the request interface that allows each client to execute their service request and to display the final result. The client submits service requests to the cloud via cloud manager for processing.

- Cloud Manager Layer: has a very important role. It is installed in the intermediary layer and is responsible for coordination between the cloud service provider and clients. Cloud Manager receives the host resource data from the monitoring module to elaborate its distribution solution. It selects the servers in each cluster that participate in calculation, and distributes the service requests for processing according to the computing power availability. Finally, it receives the results from different servers which are sending to the user.

- Cloud Resource Layer: is composed of a set of clusters of different types. Each cluster includes a different number and types of physical computing servers that provide hardware infrastructure. Client's applications will be deployed and executed in this environment.

In this work, we consider that each job is assigned to only one cluster and each cluster is composed of different number of heterogeneous computing resources or VMs (Virtual Machines). After clients submit their jobs into the cloud computing environment through the service request manager, Cloud controller divides them into several independent tasks. The task scheduler aims at determining a suitable assignment of resources to job tasks to complete all jobs received from clients.

In our model, the aim of task scheduling is to execute N Jobs on M clusters efficiently, the result of which should satisfy clients' expectations. Each Job is decomposed into several independent tasks $J_i=\{T_{i1}, T_{i2}, \ldots, T_{in}\}$ that are assigned to a set of resources in the same cluster $C_j=\{R_{j1}, R_{j2}, \ldots, R_{jn}\}$.

We assume that each cluster can estimate how

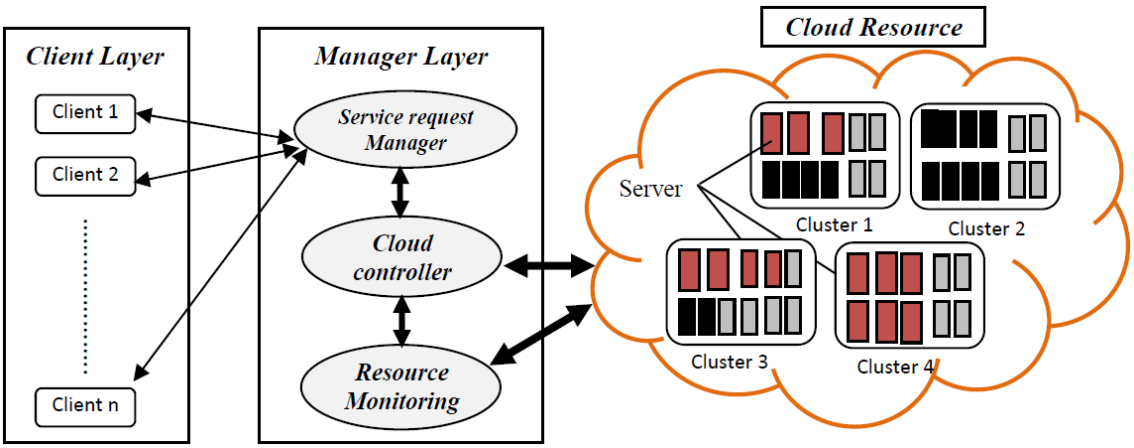


Figure 1: Cloud computing Architecture considered in this problem.

much time is required to perform each job. Expected Time to Compute (ETC) is used to estimate the expected execution time of job J_i on the cluster C_j. ETC is an $n \times m$ matrix; where n is the number of jobs and m is the number of clusters. The number of resources for each cluster is denoted R_k. The objective is to obtain an optimal task mapping on resources by minimizing completion time of tasks (makespan) while maximizing the use of resources.

The makespan is equal to maximum completion time among all tasks and can be estimated using the following equation:

$$Makespan = \max_{j \in \{1,\ldots,m\}} \sum_{i \in J_{n_i}} ETC_{ij} * B_{ij}$$

where J_{ni} is the set of jobs assigned to cluster C_j, and B_{ij} is a pseudo-Boolean integer used to determine whether the job of i^{th} client is assigned to the j^{th} cluster :

$$B_{ij} = \begin{cases} 1, \textit{ if the job of } i^{th} \textit{ client is assigned to the } j^{th} \textit{ cluster} \\ 0, \textit{ otherwise} \end{cases}$$

In order to maximize the use of resources, we have used a new parameter $\overline{R}$ that represents the average number of resources used and which is defined as the sum of all resource utilizations in each cluster divided by the total number of resources R :

$$\overline{R} = \sum_{j=1}^{m} R_j / R$$

Subject to:

- i represents the client index

$$\sum_{j=1}^{m} \sum_{k \in C_j} B_{ij} a_{ik} = 1$$

$$\sum_{i=1}^{n} B_{ij} p_{ik} \leq 1$$

$$\sum_{j=1}^{m} B_{ij} = 1 \qquad \forall i$$

α_{ik} : *The tasks of the i^{th} client assigned to the k^{th} resource*
P_{ik} : *The processing capacity of the k^{th} resource allocated to the i^{th} client*

3.2 Proposed Scheduling Approach

The problem of task scheduling is a challenge in cloud environment that aims at minimizing the completion time of tasks and maximizing the use of resource processing capabilities. Task scheduling strategy takes into account the processing requirements for each task and the mapping mechanism of these tasks according to the processing capabilities of available resources.

A heuristic-based load balancing solution is presented for task scheduling problem in cloud computing environment. The proposed algorithm uses an initialization algorithm in its first step to schedule all tasks on the available resources. The principle of the proposed approach consists of swapping the tasks between two machines showing the maximum and minimum execution times in order to minimize the makespan. Then, in the second step, we generalize this exchanging technique over all machines until the makespan value does not change, as shown in Figure 2. The same algorithm is used to assign jobs on clusters.

4 EXPERIMENTAL RESULTS

In this section, we present a set of experiments realized by CloudSim simulator to assess the proposed heuristic with respect to known task scheduling algorithms, like Min-min, in cloud computing environment. Two series of simulations have been carried out: the first uses ETC matrices randomly generated by the CloudSim Simulator, whereas the second series uses reference data with 12 different types of ETC matrix up to 16 and 32 heterogeneous machines, and up to 512 and 1024 randomly generated heterogeneous tasks.

For each randomly generated situation, the first step of the proposed algorithm consists in assigning tasks to resources in order to find a good initial scheduling. The aim of the second step is to improve the initial solution. Hence, the aim of the load balanced approach is twofold: (i) to decrease the makespan; and (ii) to increase the resources utilization. Table 1 shows the obtained comparison results between the proposed approach and the Min-Min algorithm. We used for these experiments, ETC matrices of 10 machines and up to 100 and 900 tasks.

In the second series of simulations, we show the interest of the proposed load balancing strategy with respect to the Min-Min heuristic. All experimental results represent the average makespan over 10 different *ETC* matrices of the same property. Figure 3 shows the makespan distribution for 12 different types of ETC matrix up to 32 heterogeneous machines, and up to 1024 randomly generated heterogeneous tasks (all heterogeneity cases). Table 2 shows the comparison results between the proposed heuristic and the Min-min algorithm on makespan value for 12 heterogeneity cases.

// Initialization step: Assign task to the resource

1) Determinate the available performance for all resource
2) Compute the Execution Time (ETC) for each task on all resource
3) Sort the tasks in ascending order of their ETC
4) Assign tasks to the resources with gives minimum execution time
5) Find the initial tasks scheduling matrix M

// Improvement heuristic

Select the resource R_j which fives the maximum completion time C_{max} from M
while C_{max} decrease ***do***
for all assigned tasks T_i to the resource R_j ***do***
for all $R_{j'} \neq R_j$ ***do***
for all assigned tasks T_k to the resource $R_{j'}$ ***do***
swap task T_i with T_k
update $C_{ij'} = r_{j'} + ETC_{ij'} - ETC_{kj'}$
update $C_{kj} = r_j + ETC_{kj} - ETC_{ij}$
if $(C_{ij'} < C_{max})$ and $(C_{kj} < C_{max})$ ***then***
confirm the swapping of task T_i with T_k
update M
update C_{max}
end
end
end
end
end
// C_{ij}: *expected completion time of* T_i *on resource* R_j;
// r_j: *expected time of resource* R_j;
// ETC_{ij}: *compute execution time of task* T_i *on resource* R_j;

Figure 2: Proposed heuristic algorithm for Task scheduling.

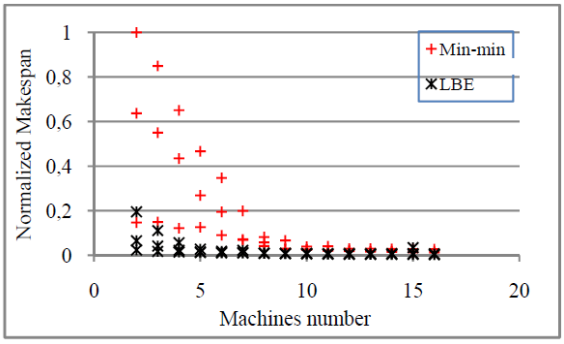
Instances are labeled as *Tt-Mm-Cc* as follows: *T* indicates the tasks heterogeneity (l: low and h: high), *M* represents the machine heterogeneity and *C* shows the type of consistency; ***c***: consistency; ***i***: inconsistency and ***s***: semi-consistency. In this work, we consider that the most important criterion is minimizing the makespan value to measure the quality of schedule.

Table 1: Simulation results of makespan for min-min and proposed algorithm.

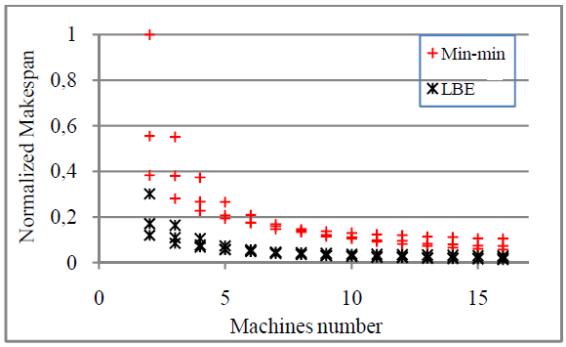
Number of Cloudlets	Min-Min Algorithm	Proposed algorithm
100	4879	4285
200	10485	5017
300	15102	7242
400	21476	10882
500	24950	12185
600	26489	13752
700	29225	16537
800	34896	23394
900	33648	27548

Figure 3 shows also the makespan distribution for 12 different types of ETC matrix up to 32 heterogeneous machines, and up to 1024 randomly generated heterogeneous tasks (all heterogeneity cases). Each of the four heterogeneity cases has means of {Min-Min (0.1311), LBE (0.004)}, {Min-Min (0.0129), LBE (0.007)}, {Min-Min (0.0209), LBE (0.0111)} and {Min-Min (0.1294), LBE (0.0219)}. Their standard deviations are {0.1210, 0.005}, {0.1358, 0.014}, {0.1114, 0.0137} and {0.0994, 0.0252} respectively.

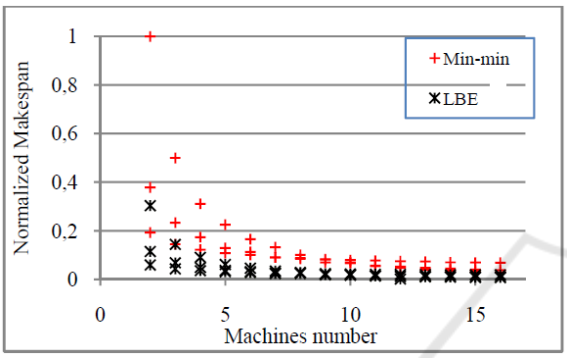
Clearly for all heterogeneity cases, the points of the makespan distribution obtained by the Min-Min heuristic are completely dispersed over the points of the makespan values obtained by our algorithm for all machines. The completion times obtained by the Min-min heuristic have a large standard deviation which indicates that the completion times of the machines are far from the mean. Our completion time has a much smaller standard deviation than the other two because its values are all close to the mean. This indicates that the completion times are clustered closely around the mean and the refined of the completion time values are very close.



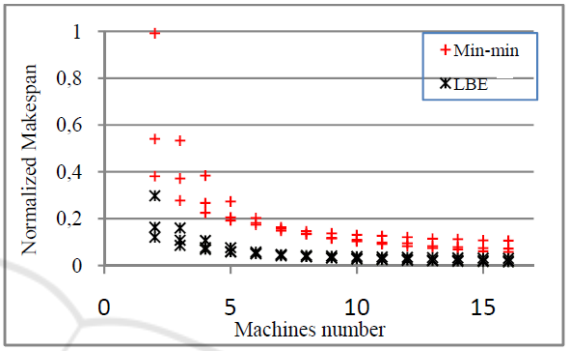
a. Low task and low machine heterogeneity cases.



b. Low task and high machine heterogeneity cases.



c. High task and low machine heterogeneity cases.



d. High task and high machine heterogeneity cases.

Figure 3: Makespan values distribution for different heterogeneity cases.

Table 2: Comparaison of makespans results.

Instance	Min-Min algorithm	Min-Min + AG	Proposed algorithm
ThMhCc	9267258	9219257	**9173984**
ThMhCi	17457713	1463582	**10645286**
ThMhCs	16458962	1256743	**10548555**
ThMlCc	564889	250384	**181867**
ThMlCi	143908	129983	**121159**
ThMlCs	744679	572668	**446259**
TlMhCc	690435	398865	**315852**
TlMhCi	3097331	161032	**130576**
TlMhCs	1346356	521061	**434871**
TlMlCc	18738	6752	**5963**
TlMlCi	4306	3971	**3430**
TlMlCs	23868	18043	**12590**

Experimental results reported in Table 2 show that under high task and machine heterogeneities, our algorithm improves the makespan from 1% to 39% in all consistency situations. However, in low task and high machine heterogeneities, our algorithm gives better quality scheduling solutions between 54.25% and 95.78% than Min-min algorithm in all consistency cases. This indicates that the proposed scheduling algorithm performs well compared with Min-Min heuristic scheduling algorithm in terms of performance and makespan (improvements between 15..80% and 95.78%) except in the consistent case for high tasks and machines (1%).

5 CONCLUSIONS

In this paper, we have proposed a new task scheduling heuristic based on makespan improvement in cloud computing environment. The proposed approach generalizes the process of tasks exchange between both machines with the maximum and minimum makespan over all machines until the makespan value does not change. Our approach has then been assessed with respect to the Min-min algorithm. The results obtained have shown significant improvements in terms of execution time, makespan and resources utilization. An interesting future direction would be to evaluate our proposed solution against other State-of-the-Art task scheduling algorithms.

REFERENCES

Abirami, S.P. and Shalini, R. 2012. Linear Scheduling Strategy for Resource Allocation in Cloud Environment. International Journal on Cloud Computing: Services and Architecture (IJCCSA), Vol.2, No.1, February 2012.

Beghdad-Bey, K., Benhammadi, F., Sebbak F. and Maataoui, M. 2015. New Tasks Schedeling Strategy for resources Allocation in Cloud Computing Environment. Sixth International Conference on Modeling, Simulation and Applied Optimization, ICMSAO'2015, May 27-29, 2015.

Delhi Babua, K. and Giridhar Kumar, D. 2014. Allocation Strategies of Virtual Resources in Cloud-Computing Networks. International Journal of Engineering Research and Applications, Vol. 4, Issue 11, November 2014, pp.51-55.

Fang, Y., Wang, F. and Ge, J.2010. A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing. Web Information Systems and Mining, Lecture Notes in Computer Science, Vol. 6318, 2010, pages 271-277.

Gomathi, B. and Karthikeyan, K. 2013. Task scheduling algorithm based on hybrid Particle swarm optimization in Cloud Computing environment. Journal of Theoretical and Applied Information Technology, September 2013. Vol. 55 N° 1.

Gouda,K. C., Radhika, T. V. and Akshatha, M. 2013. Priority based resource allocation model for cloud computing. International Journal of Science, Engineering and Technology Research (IJSETR), Volume 2, Issue 1, January 2013.

Goudarzi, H. and Pedram, M. 2011. Maximizing Profit in Cloud Computing System via Resource Allocation. IEEE 31st International Conference on Distributed Computing Systems Workshops 2011: pp, 1-6.

Guo, L. 2012. Task Scheduling Optimization in Cloud Computing Based on Heuristic Algorithm", Journal of Networks, vol. 7, NO. 3 march 2012.

Irugurala, S. and Chatrapati, K.S. 2013. Various Scheduling Algorithms for Resource Allocation. In Cloud Computing. The International Journal of Engineering And Science (IJES), Volume 2, Issue 5, Pages 16-24, 2013.

Jeyaram, G., and Vidhya, V., 2013. Efficient resource allocation strategies for cloud data centers. International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 2, February 2013.

Kanrar, S. 2012. Enhancement of job allocation in private Cloud by distributed processing. Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology (CCSEIT'12), Pages 94-98, ACM NY 2012.

Katyal, M. and Mishra, A. 2014. Application of Selective Algorithm for Effective Resource Provisioning In Cloud Computing Environment. International Journal on Cloud Computing: Services and Architecture (IJCCSA), Vol. 4, No. 1, 2014.

Keshk, A. E, 2014. Cloud Computing Online Scheduling. IOSR Journal of Engineering (IOSRJEN), Vol. 04, Issue 03, March. 2014.

Kumar, P. and Verma, A. 2012. Scheduling Using Improved Genetic Algorithm in Cloud Computing for Independent Tasks. Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI '12), Pages 137-142, ACM, NY 2012.

Kuribayashi, S. 2011. Optimal Joint Multiple Resource Allocation Method for Cloud Computing Environments. International Journal of Research and Reviews in Computer Science (IJRRCS), Vol. 2, No. 1, March 2011.

Selvarani, S. and Sadhasivam, G.S. 2010. Improved cost-based algorithm for task scheduling in Cloud computing. Computational Intelligence and Computing Research (ICCIC), IEEE, pp.1-5, 2010.

Silva, J.N., Veiga, L. and Ferreira, P. 2008. Heuristic for resources allocation on utility computing infrastructures. MGC'08 Proceedings of the 6th International Workshop on Middleware for Grid Computing, ACM, NY, USA, 2008, pp. 1–6.

Tawfeek, M., El-Sisi, A., Keshk, A. and Torkey, F.2015. Cloud Task Scheduling Based on Ant Colony Optimization. International Arab Journal of Information Technology, Vol. 12, No. 2, 2015.

Vinothina, V., Sridaran, R. and Ganapathi, P. 2012. A Survey on Resource Allocation Strategies in Cloud Computing. International Journal of Advanced Computer Science & Applications, 2012, vol. 3, n°6.