A Nature-Inspired approach for Load Balancing of tasks in cloud computing using Equal Time Allocation

B. Mallikarjuna, P. Venkata Krishna

Abstract: Cloud computing is treated as one of the efficient paradigm for distributed computing. Due to the vast development in the internet world, scheduling of resources on cloud computing makes the issues. Load balancing over a virtual machines place a major role in efficient scheduling. In this paper, we propose a mechanism for load balancing among virtual machines by using a nature inspired approach called as honey bee foraging (LB-HBF). The task and virtual machine allocation mechanism which is used inside this LB-HBF is equal time allocation policy. This policy works efficiently for allocating the tasks to the virtual machines and reducing the task migrations over virtual machines. The experimental results showed the effectiveness of LB-HBF.

Key words: load balancing, tasks, virtual machines, honey bee, resource management.

I. INTRODUCTION

Cloud computing is treated as one of the advanced development in distributed computing, it is developed for the business rather than academic which shifts its focus towards user applications [1].These cloud computing offers virtualized, distributed and elastic resources as services to the end users. It has a great edge to support full realization of "computing as a service" in future. The cloud provides the privileges to the users to use the computing power of the services by renting policy. These processesare carried on virtualization concept, where it can have hundreds and thousands of virtual machines (VMs) which can provide services to the users. Manually resource allocation is not possible in a cloud environment, so we rely on virtualization concept.

One of the major issues related to virtualization is load balancing. The major studies are carried out in the field of load balancing problem, but still cloud computing makes it an interesting topic and many research projects are under way [2]. This is due to the typical nature of cloud computing and the characteristics of the problem itself. The classical load balancing algorithms can be applied only for homogeneous and dedicated resources, cannot work well for the cloud computing [3]. Cloud computing has a lot of special characteristics, like heterogeneity, dynamicity and autonomy, which are not possible to apply classic load balancing algorithms directly to the cloud computing.

Load balancing is defined as the process of distributing the tasks equally over the computing resources in the data centers to improve the performance in cloud computing. The essential objective of the load balancing can be dependent on the user or the service provider, defined by:

- The objective of the user is to reduce the makespan of its own application, regardless of the other applications in the system.
- The objective the service provider is to speed up the task completion time and effective utilization of available resources.

A realistic approach to dynamic load balancing is to divide the problem into following four policies.

1) Calculation of load: some estimation of VMs load must be provided to first calculate the load imbalance exists. Estimation of workloads includes with individual tasks must be maintained to identify which task should migrate for balance the computation.

2) Initiation of load balance: once the loads of all VMs are calculated, if there is any imbalance is detected. Then the cost of the load imbalance exceeds the cost of load balance, and then load balancing is initiated.

3) Selection of task: the tasks are selected to transfer from one VM to another VM based on the information provided by the above steps.

4) Migration of tasks: after selecting the tasks, transfer of tasks is initiated from one VM to another VM. Algorithm correctness must be maintained in the above steps.

In this paper, we propose a nature inspired approach based on honey bee load balancing algorithm for the allocation of tasks to the virtual machines in the context of federated clouds [4]. Our model suggests the hypothetical colony whose foragersfollowstheforagingbehaviour of honey bees for load balancing of VMs in cloud computing. We find that the model is suitable for the allocation of tasks among VMs.

The rest of the paper is organized as follows. Section 2 deals with the previous work which carried over. Section 3 deals with the problem formulation and section 4 deals with the load balancing mechanism which includes equal allocation policy among VMs. Section 5 deals with analysis of simulation environment and finally concludes the results in section 6.

II. RELATED WORK

In [5] the graph algorithm is used to solve the load balancing issues for resource allocation. However, the algorithm doesn't address the minimization of cost,i.e., the

Revised Manuscript Received on December 08, 2018.

B. Mallikarjuna, Assistant Professor, SCSE, Galgotias University, Greater Noida, Uttar Pradesh, India

P. Venkata Krishna, Professor & Director, Dept of CS, SPMVV, Tirupati, Andhra Pradesh, India.

cost handled at the load redistribution, which can take more time than the actual time consumed for the computation.Some study [6] proposed latency algorithms to reduce the migration cost for computation of internal data and taking the advantage of reducing the data in communication. But this type of algorithms needs some type of parallel applications to compute the data processing and migration.

In [7], it introduces the master slave load balancing to optimize the data distribution and migration using the linear programming algorithm. But this algorithm only deals with static load balancing. In [8], it proposed an optimal data migration algorithm for dynamic load balancing based on the calculation of Lagrange multiplier for the Euclidean form of the transferred weight. But it works only on homogeneous environment and it does not work on heterogeneous networks.

The objective of the load balancing is to balance the computational tasks over the virtual machines. The balancing of computation tasks to minimize the execution time is called makespan. Moreover, makespan minimization problem is common in distributed systems; we commonly called as NP-complete [16][17]. In regards this, minimization of makespan is not only the task of load balancing, but also it has to deal with the communication cost and load redistribution is also a major issue.

Some of the works related to load balancing in cloud computing based on bio inspired approaches. In [9] a random key genetic algorithm is applied for hybrid cloud services. Other works [10], [11] uses nature inspired approaches for submitting the tasks in grid computing.

III. PROBLEM FORMULATION

The cloud computing environment associated with the thousands of tasks which are submitted by the users. Thesetasks are divided and assigned to the specific data centers where it has to be computed. The data centers are associated with the number of VMs to execute the tasks which are allocated by the resource broker. The variables which are used in this model is shown in table 1.

Table 1: variables used in this model

variables	Definitions
ϑ_x	No. of tasks executing at the virtual machines
η	Total no. of Active VMs
δ_x	Rate at which the completion time of each task
φ_x	Waiting time of each task for virtual machine
ω_x	Average waiting time of tasks for virtual machine
τ_x	Total completion time of tasks at VMs

Let us assume that nx be the number of tasks executing at the VMs, and $\delta x(\vartheta x)/\vartheta x$ is the average executing time for all tasks on a VMs in a unit time, is a non-increasing function of ϑx, the number tasks allocated to the VM.Here waiting time φx is dependent on the nx, but this dependent is in terms of some quantity (i.e.The efficiency of the virtual machine) the individual task can access. For simplification, we write φx rather than $\varphi x(\vartheta x)$. If any virtual machine is available without allocating any task that will be advertised by using waggle dance. The average duration for the dance for virtual machine x is given as γx, then the probability for allocating the task to the virtual machine x is given as $\gamma x \delta x(\vartheta x)$, the total amount of dancing time for the virtual machine x in a fixed period of time, rather than the total number of dances for the virtual machine.Againγx is a function of ϑx, but the dependency between γx and ϑx must be measured in terms of individual tasks it handles.

IV. LOAD BALANCING USING HONEY BEE FORAGING (LB-HBF) BEHAVIOR

In this paper, we are going to organize the effective allocation of tasks for VMs can be carried by using a bio-inspired approach called as honey bee foraging behavior (HBF). In this HBF the honey bees are represented in finding the food and reaping the food. There is a colony of bees which forages for the food resources. After finding the food resources the bees make one type of signature dance called as waggle dance. They came back to the hive to advertabout the food by making waggle dance. Based on the duration of the dance, gives the idea about the quantity of food and distance from the bee hive. The scout bees follow the foragers to the place where the food is available and then began to reap it.

In the case of load balancing mechanism for cloud computing, the demand for virtual machines increases or decreases, the services of data centers are assigned dynamically based on the changing demands of the users. The data centers consist of a group of virtual machines having their own virtual machine queues. Each virtual machine processes a request from its queue based on the profit, which is identical with the quantity of the waggle dance showed by bees[18][19]. The measuring of the profit can be calculated from the task execution time at virtual machine. The dance floor in the case of honey bees is identical to the advert board. This board is used for the display of whole profit of the colony or group of VMs. Each of the virtual machine can take the place of forager or scout. The process of allocation of virtual machines based on waggle dance is shown in figure 1.

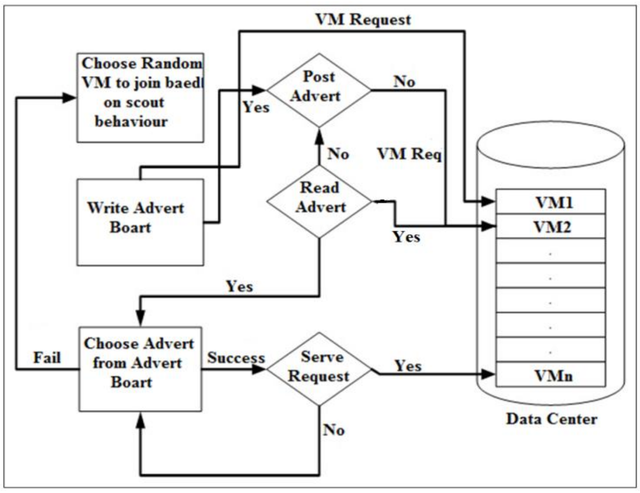

Figure 1: Load balancing over VMs using HBF adapted from [12]

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Algorithm 1: Virtual Machine Behavior in the HBF

```
Begin
Initialize x µservingQj, post advert probability p,
 Advert reading probability ri, , completion time τ
forever
whileQj != empty do
serve request
if task is completed then
Compute completion time τ;
adjustri from the queue;
if Write(p)== TRUE then  post advert;
if task is completed and read(ri)==TRUE then
  /* randomly select the Virtual machine or a task */
  Select advert (if task) or data center (if virtual machine);
Read advert id µj (if virtual machine);
end if;
end if;
end while;
end forever;
end
```

The behavior of any virtual machine in the data center is controlled by algorithm 1. A virtual machine x µ is a set of virtual machines from service request queue Qj will prompt the probability p to post the advert board.Also, it will attempt with probability ri to read a randomly select advert from advertboard if it is a task or a virtual machine.

4.1 Mathematical model for LB-HBF

To understand the load balancing approach in cloud computing based on honey bee foraging behavior, we are going to develop a mathematical model supports the differential equations which models the all allocation mechanisms in VMs colony. The differential equations approximate our colony model. First, we model the foraging activity of virtual machines, second, for a given set of virtual machines there is a unique allocation of tasks to which the process will converge[20]. The allocation process in the cloud is stochastic: it will not follow any single approach for an extended period of time.But, it will approach a steady state allocation.

As a first step, we consider all dances at the same quantity, i.e., all virtual machines are at equal load in our colony. The colony of virtual machines is denoted with µ. When all loads are at the same quantity, an allocation and leads to advert for virtual machine x at the rate of execution is given as δx (ϑx) and so to advert for all VMs at the combined rate of

$$\chi(\vartheta) = \sum_{x\in\mu} \delta_x(\vartheta_x) \qquad (1)$$

Thus, of all tasks re-entering the allocation process during a sufficient period of time, the fraction

$$\frac{\delta_x(\vartheta_x)}{\chi(\vartheta)}$$

are applied to virtual machine x on average

We assume that the number of active VMs is fairly constant over a long period of time, hovering about the value η, so for each allocation ϑ,

$$\eta = \sum_{x\in\mu} \vartheta_x \qquad (2)$$

To achieve this, the tasks must re-enter into the allocation process at the same average rate they are completed from it.

Thus, tasks re-entering the allocation process at the rate

$$\bar{\mu}\eta \frac{\delta_x(\vartheta_x)}{\chi(\vartheta)} \qquad (3)$$

and the new tasks are recruited to virtual machine x at the rate which is shown in equation 3.

Since each task is completed at the rate

Then the rate at which the task completion time for virtual machine x is given as

$$\bar{\mu}\vartheta_x \qquad (4)$$

By combining the equation 3 and 4, we get the differential equations about how the allocation process will change with time:

$$\vartheta_x = \bar{\mu}\eta \frac{\delta_x(\vartheta_x)}{\chi(\vartheta)} - \bar{\mu}\vartheta_x \qquad (5)$$

The model reaches a steady state allocation in which task allocation and completion done at the same rate, so the number of tasks allocated to each virtual machine remains constant and

$$\frac{\vartheta_x}{\delta_x(\vartheta_x)} = \frac{\eta}{\chi(\vartheta)} \qquad (6)$$

for each active virtual machine x.

The total execution time of each task τ at x is given as the ratio $\frac{\vartheta_x}{\delta_x(\vartheta_x)}$,

Although the differential equation 5 simplifies the allocation process in our colony model, both are simulated up to their dynamics. If the total execution time of the task at virtual machine x is too large, the average allocation time drops. As this rate drop, it reflects on the average advert time on the virtual machine and allocates migrates to another virtual machine. This means at each virtual machine can allocate to the newly arrived task to take the place at a slower than average rate[21]. Thus the number of new tasks directed to the virtual machine is smaller than the number of tasks diverted from it and the number of tasks allocated to the VMs declines. The rate of the tasks declines, so, the average completion time. Similarly, if the average completion time for a virtual machine is too small, the larger the advert time for that virtual machine will increase the number of task allocations and within the average completion time.

The fact, that our allocation method reaches a steady state allocation by using differential equation 5. In order to better understand this method; we consider the colony after a long period of time. In terms of the differential equation 5 this means that the colony tends to a steady state allocation ϑ active on a set µ| of virtual machines with x in µ| if and only if for small ε>0.

$$\vartheta_{x|\varepsilon} = \bar{\mu}\delta_x(\varepsilon)\left[\frac{\eta}{\chi(\vartheta)} - \frac{\varepsilon}{\delta_x(\varepsilon)}\right] \geq 0 \qquad (7)$$

That is, if and only if the average completion time $\frac{\varepsilon}{\delta_x(\varepsilon)}$

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

on virtual machine x with ε tasks is less than the average completion time of overall tasks τ.In simple, the advertis the same duration and all tasks completed at the same average rate irrespective of what virtual machine it is going to be visiting. We refer this as the equal time allocation policy.

Theorem 1: let ϑ be any allocation and ϑ* be the equal time allocation. If at each virtual machine x, δx is non-decreasing and $\delta_x/\vartheta_x$is non-increasing, then $\frac{\chi(\vartheta)}{\chi(\vartheta^*)} \leq 2.$

Proof of Theorem 1:

Consider any task allocation ϑ. Let μ<= be the set of virtual machines x for which ϑ_x<=ϑ_x* and let μ> be the set of virtual machines x for which n_x>n_x*. then

$$\sum_{x\in\mu_{\leq}} \delta_x(\vartheta_x) \leq \sum_{x\in\mu_{\leq}} \delta_x(\vartheta_x^*) \leq \chi(\vartheta^*) \qquad (8)$$

Since δx is non-decreasing. Further let

be the common average completion time of the active virtual machines under the equal time allocation. Since δ_x/ϑ_x is non-increasing and ϑ_x*=0 only if

ε/δ_x(ε)> for small ε>0,

$$\frac{\delta_x(\vartheta_x)}{\vartheta_x} \leq \frac{1}{\tau}$$

For each x μ>.Thus,

$$\sum_{x\in\mu_{>}} \delta_x(\vartheta_x) \leq \frac{1}{\tau}\sum_{x\in\mu_{>}} \vartheta_x \leq \frac{\eta}{\tau} = \chi(\vartheta^*) \qquad (9)$$

and, by combining equation 8 and 9 gives the result

$$\chi(\vartheta) = \sum_{x\in\mu} \delta_x(\vartheta_x) \leq 2\chi(\vartheta^*) \qquad (10)$$

Theorem 1 states that even with complete information about the δx, perfect communication ad unlimited power, it is the most efficient model and it is not possible to find an allocation more than twice as efficient as the equal time allocation.

V. RESULTS EVALUATION

In this section we deal with the load balancing of virtual machines.We developed a simulation environment for a honey bee foraging model for load balancing of virtual machines and comparison can be done with the omniscient, greedy and optimal-static algorithms. The simulation environment is tested with a trusted toolkit called as cloudsim[13-15]. This simulation environment gives the change to design, implement and test the cloud computing infrastructure. In this simulation, we have tested the algorithms with different low and high loads and calculated the makespan of the algorithms.

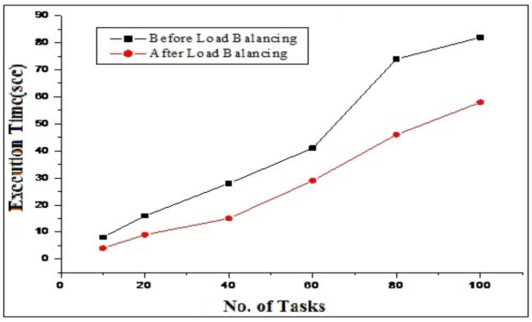

Figure 2: comparison of makespan before and after applying LB-HBF

Figure 2 illustrates the comparison graph between before and after applying LB-HBF, here we observed that execution time is improved in case of LB-HBF. In figure 3 we observe the improvement in response time in VMs of LB-HBF when compared to other three algorithms.

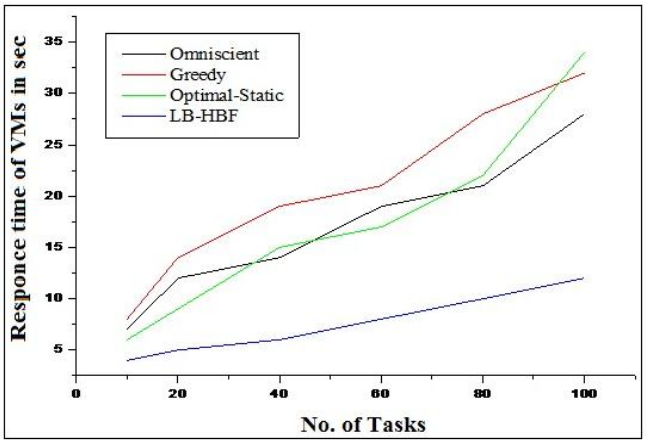

Figure 3: Response Time of VMs in omniscient, greedy, optimal-static and LB-HBF

In Figure 4 it shows the makespan of the four algorithms implemented over 100 tasks and the improvement is identified in LB-HBF.

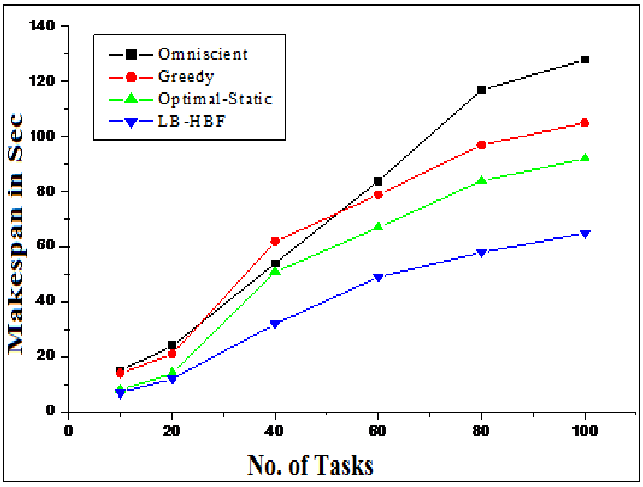

Figure 4: Makespan of omniscient, greedy, optimal-static and LB-HBF

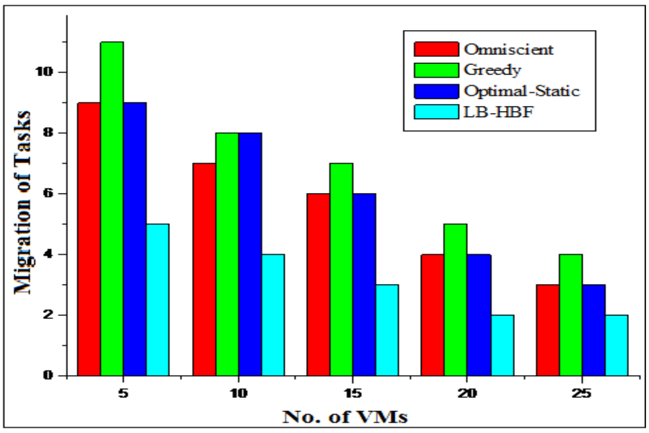

Figure 5: number of task migrations over VMs when tasks=20

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

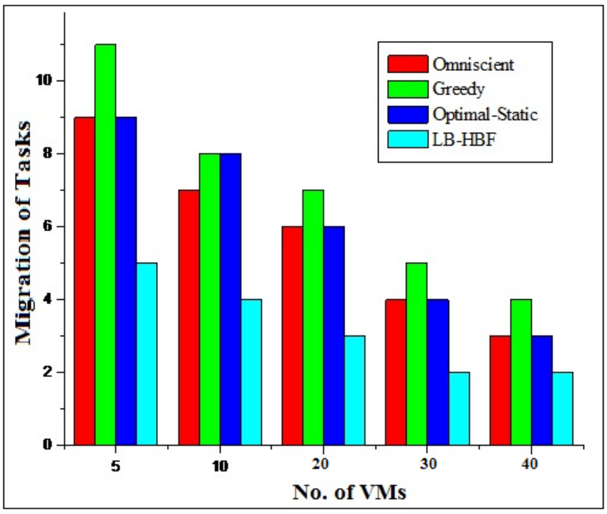

Figure 6: Number of task migrations over VMs when tasks=40

Figures 5 and 6 represent migration of tasks over VMs which is tested over 20 and 40 tasks. This will clearly shows that the task migrations in LB-HBF is smaller when compared to the other three algorithms.

VI. CONCLUSION

In this paper, we are concentrate on load balancing of tasks using a nature-inspired approach called as honey bee foraging(LB-HBF).This mechanism works with the principle of honey bees.Here we introduced a new policy for allocation called as equal time allocation among tasks and virtual machines.This allocation policy decreases the migration cost of tasks and improved the efficiency of the model.The LB-HBF works well in reducing the load balance over VMs, increasing the response time of VMs and reducing the makespan. The results shown the efficiency of the alagorithm. In future we are planning to investigate the different nature inspired approaches over the load balancing problem for cloud computing.

REFERENCES

1. Gon¸calves, G. E. et al., Resource Allocation in Clouds: Concepts, Tools and Research Challenges. XXIX Brazilian Symposium on Computer Networks and Distributed Systems, 2011.
2. Buyya, R.; Ranjan, R.; Calheiros, R.N.; Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities, High Performance Computing & Simulation, 2009. HPCS '09. International Conference on, pp.1-11, 2009.
3. Ai, L., Tang, M. and Fidge, C. QoS-oriented Resource Allocation and Scheduling of Multiple Composite Web Services in a Hybrid Cloud Using a Random-Key Genetic Algorithm, Australian Journal of Intelligent Information Processing Systems, Vol. 12, No.1, 2010.
4. Buyya R., Ranjan, R., Calheiros, R. N. InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing, LNCS, Springer, Germany, 2010.
5. H. Johansson and J. Steensland, "A performance characterization of load balancing algorithms for parallel SAMR applications," Uppsala University, Department of Information Technology, Tech. Rep. 2006-047, 2006.
6. H. Shan, L. Oliker, R. Biswas, and W. Smith, "Scheduling in heterogeneous grid environments: The effects of data migration," in Proceedings of ADCOM2004: International Conference on Advanced Computing and Communication, India, December 2004.
7. S. Genaud, A. Giersch, and F. Vivien, "Load balancing scatter operations for grid computing," in Proceedings of the 12th Heterogeneous Computing Workshop (HCW'2003), Nice, France, April 2003, pp. 101–110.
8. Y. Hu, R. Blake, and D. Emerson, "An optimal migration algorithm for dynamic load balancing," Concurrency: Practice and Experience, vol. 10, pp. 467–483, 1998.
9. Ai, L., Tang, M. and Fidge, C. QoS-oriented Resource Allocation and Scheduling of Multiple Composite Web Services in a Hybrid Cloud Using a Random-Key Genetic Algorithm, Australian Journal of Intelligent Information Processing Systems, Vol. 12, No.1, 2010.
10. Song, S., Hwang, K., Kwok, Y., "Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling,"Computers, IEEE Transactions on , vol.55, no.6, pp.703-719, 2006.
11. Entezari-Maleki, R. andMovaghar, A. A Genetic-Based Scheduling Algorithm to Minimize the Makespan of the Grid Applications. Grid and Distributed Computing, Control and Automation. Springer, 2010.
12. Ratan Mishra and AnantJaiswal, "Ant colony Optimization: A Solution of Load balancing in Cloud," InternationalJournal of Web & Semantic Technology Vol.3, No.2, pp33- 50,2012.
13. R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A.F. De Rose, R. Buyya, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software: Practice and Experience 41 (2011) 23–50, http://dx.doi.org/10.1002/spe.995.
14. R.N. Calheiros, R. Ranjan, C.A.F.D. Rose, R. Buyya, CloudSim: a novel framework for modeling and simulation of cloud computing infrastructures and services, Computing Research Repository, vol. abs/0903.2525, 2009.
15. R. Buyya, R. Ranjan, R.N. Calheiros, Modeling simulation of scalable cloud computing environments and the cloudsim toolkit: challenges and opportunities, in: Proceedings of the 7th High Performance Computing and Simulation Conference (HPCS 2009, ISBN: 978-1-4244-4907-1, IEEE Press, New York, USA), Leipzig, Germany, June 21–24, 2009.
16. T. S. K. Reddy, P. V. Krishna and P. C. Reddy, "Power Aware Framework for Scheduling Tasks in Grid based Workflows", Int. J. Communication Networks and Distributed Systems, Inderscience Publishers, 2014 (accepted)
17. L.D.DhineshBabu, AngappaGunasekaranand P Venkata Krishna," A Decision Based Pre-emptive Fair Scheduling Strategy to Process Cloud Computing Work-flows for Sustainable Enterprise Management", International Journal of Business Information Systems, Vol. 16, No.4, pp. 409-430, Inderscience Publishers, 2014
18. Babu, L. D., & Krishna, P. V., "An execution environment oriented approach for scheduling dependent tasks of cloud computing workflows", International Journal of Cloud Computing, 3(2), 209-224, 2014.
19. L.D.DhineshBabu and P VenkataKrishna,"Honey Bee Behavior inspired Load Balancing of tasks in Cloud Computing Environments", Applied Soft Computing, Volume 13 Issue 5,
20. Pages 2292-2303 Elsevier, May, 2013.
21. Krishna, P.V.; Misra, S.; Joshi. D.; Obaidat, M.S.; Learning Automata Based Sentiment Analysis for Recommender System on Cloud, 2013 IEEE International Conference on Computer, Information, and Telecommunication Systems, CITS 2013, May 7-8, 2013
22. PriyaPandey, P Venkata Krishna, B Sarojini, "QoS Aware Healthcare System on Mobile Clouds", 2014 World Congress on Computing and Communication Technologies (WCCCT), 154-157, IEEE, 2014