IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-10, Oct 2018 E-ISSN: 2347-2693

LAHUBMAX –Priority Based Meta Task Scheduling Algorithm in multicloud

BJ. Hubert Shanthan[1*], L. Arockiam[2]

[1,2]Dept. of Computer Science , St.Joseph's College, Trichy, India

[*]*Corresponding Author: hshanthan@gmail.com*

Available online at: www.ijcseonline.org

Accepted: 10/Oct/2018, Published: 31/Oct/2018

***Abstract*—** Cloud Computing plays a obvivious role in the field of web based computing.Multicloud is an advancement of the cloud computing. Multicloud is an environment comprises of more number of cloud service providers available to cater the needs of the heterogenous users. Task scheduling is one of the major issue affects the performance of the multicloud computing systems. This paper is designed to solve task based scheduling problem for the multicloud systems. Priority is one of the key concern to the service consumers. The high prioritized tasks are given due importance and they are executed in the high speed virtual machines. This article proposes a novel priority based independent task based scheduling algorithm for the multicloud environment. Fuzzy rule is used in this algorithm , to select the user prioritized tasks with larger length. Mandomi Inference system to generate a new rule to select and execute tasks. The proposed algorithm is merely emphasized on minimizing the total completion time of the tasks. There are five different categories of user priorities illustrated in this algorithm. The proposed algorithms outperforms the existing max-min algorithm in terms of makespan and cost.

***Keywords*—** Max-min , makespan ,Multicloud , Mandomi

I. INTRODUCTION

Today cloud plays a vital role in provisioning resources to the users on-demand. Virtualization plays key role in the cloud to provide infrastructure and other services to the customer[1]. The CSP (Cloud Service Provider) is used to provide the reliable service to the user. Each cloud service provider has its own scheduling policy to process the user requests. So it is difficult to satisfy the customer needs by a single service provider. Vendor lock-in is one of the major issues in the single cloud environment[2]. In order to overcome this conflict multicloud is introduced to the needs of the customer[3].In multicloud [4] one or more service providers are used to minimize the risk of the data loss and reduce the localized component failure in a cloud. Multicloud model comprises of three components namely cloud user, cloud manager and cloud service provider. Cloud user is the client who sends the request to the service provider. Cloud manager acts as an interface between cloud user and cloud service provider. Cloud manager plays the role as a resource broker, to allocate and manage the user requests to the service provider. The cloud user requests are represented as tasks. The main job of the cloud manager is used to schedule the tasks to the appropriate cloud service provider without any information loss[5].Since multicloud is a heterogeneous environment of different users and service providers, there is a need of efficient task based scheduling algorithm in the multicloud environment[6]. The unified scheduling algorithm must be designed to cater the needs of the customer in a multicloud environment. The scheduling algorithm must be designed to reduce the makespan and increase the profit of the service providers[7].

The proposed LAHUBMAX – priority aware meta tasks is compatible for the independent tasks in a multicloud environment. The LAHUBMAX algorithm comprises of two phases namely prerogative phase and scheduling phase. The prerogative phase is used to find the minimum earliest completion time of the tasks.

This phase matches and allocates the tasks to the virtual machines of minimum completion time based on the user priorities.

The rest of this paper is organized as follows. The Section 2 consists of related work with the existing literature found in the proposed work. Section 3 describes the proposed priority based meta task scheduling algorithm. The section 4 interprets the results and discussions of the proposed algorithm . Finally section 5 ends this paper with conclusions and future directions

© 2018, IJCSE All Rights Reserved

II. RELATED WORK

In this section, Kamala, et al. [11] presented a heuristic algorithm to reduce Makespan and improve utilization of resources. The algorithm combined the benefits of minimum completion time and minimum execution time of the resources. This algorithm is comprised of two phases: first phase selects the tasks with minimum execution time and the second phase calculates the tasks with the minimum completion time and then assigns the tasks to its resources. This algorithm gave priority for the smaller tasks and so there was a rapid increase of waiting time for the larger tasks.

Rajasekar, et al. [12] proposed a mechanism to improve Makespan for maximum utilization of resources for larger tasks. This algorithm outperformed Min algorithm in terms of Makespan. This algorithm had two phases: the first phase was similar to Min-Min and in a second phase, the tasks with maximum completion time were selected and assigned to their resources. Priority was given to larger tasks as it reduced the average waiting time of the resources.

M.Maheswaran, et al. [13] proposed a suffrage based Meta task scheduling algorithm to reduce the Makespan and increased the average resource utilization rate of the servers. The suffrage value was calculated by making the difference between first minimum execution time and second minimum completion time. The tasks with high suffrage value were assigned to the resources and also considered a minimum completion time of the resources. In this algorithm, maximum suffered tasks were given priority and least suffered tasks were given less importance. This algorithm gave better Makespan results than Min-Min and Max Min algorithm.Afab [14] proposed an improved Min-Min algorithm to reduce Makespan in the Meta task scheduling in grid computing. This algorithm calculated the arithmetic mean of the minimum execution time value. The mean value was used as a threshold value and tasks were allocated to the resources. This algorithm outperformed standard Min-Min and Max Min algorithm. Load balanced Opportunistic approach [15] was easier and simpler to implement than any other mechanism. This approach resulted in a poor Makespan value. It allocated the available resources to the tasks and it made the machine busy all the time. This algorithm did not consider the execution and completion time for the resources

Thomas et al. [16] proposed a static scheduling algorithm by considering task length and user priority as credit value. The credit value acted as parameter and allocated the task based on priority and duration of the task which was requested by the user.

Parsa, et al. [17] proposed a unique resource aware scheduling mechanism for the Meta tasks. The mechanism worked on the available resources for mapping of the tasks. This algorithm used Min-Min strategy for the odd and even number of resources. The Max-Min mechanism was used for mapping the tasks to its resources.

Sharma, et al. [18] suggested a heuristic based task aware scheduling mechanism for the cloud. This algorithm checked the total number of tasks that were available for mapping its resources. The odd number of tasks used Min-Min strategy to map its resources. Max-Min mechanism was used for even number of tasks to map its resources.

Kokilvani, et al. [19] proposed an efficient load balanced Min-Minscheduling algorithm to reduce Makespan and improve the resource utilization rate. Min-Minalgorithm was used in the first stage of this algorithm and the Makespan was also calculated at this stage. Makespan value was used as a threshold value to reschedule the tasks and the tasks with the heavy load of resources were reassigned to the tasks with a light load of resources. This algorithm outperformed the standard Min-Min algorithm and improved the average resource utilization rate.

Patel, et al. [20] suggested a strategy to enhance load balanced Min-Minscheduling algorithm to minimize the Makespan and increased the average resource utilization rate. This algorithm was more suitable for independent meta tasks scheduling algorithm. Makespan value was calculated by Min-Min algorithm. The tasks were sorted in non-decreasing order with execution time and value was checked with Makespan. The sorted value with maximum execution time was reassigned to the tasks with a minimum completion time of the resources and load was balanced by selecting the minimum value from the completion time of the assigned resources of the tasks.

Load balanced Opportunistic approach [20] was easier and simpler to implement than any other mechanisms. This approach had poor Makespan. It allocated the available resources to the tasks and it made the machine busy all the time. This algorithm did not consider the execution and completion time for the resources.

III. METHODOLOGY

The proposed LAKHUB-MAX is a priority meta task scheduling algorithm. The desired objective of the proposed algorithm is to minimize the total completion time of the tasks. The algorithm is more viable, for the large number of tasks.

The LAKHUB-MAX system model comprises of two phases namely User Priority Phase and Rule phase,. In the

© 2018, IJCSE All Rights Reserved

user priority phase, the users prioritize the requests to the server. The user driven requests are considered as the tasks. The second phase is the Rule phase, in this phase the user requested tasks are termed as task processing capacity (MI) Million of Instruction volume. The total Virtual Machine Resource Processing capacity is termed as Millions of Instructions per second (MIPS). The tasks with maximum processing capacity (MI) and user priority are considered before the execution of the tasks. The ETC table is generated and it is ready to execute in a cloud environment. ETC (Expected Execution time) is constructed based on the MI and MIPs of the virtual machine.

The new priority is set to the tasks based on the fuzzy inference rule. Mandomi Defuzzification centroid model is used to get the output results from the crispy inputs.The user priority are given as inputs. MCT (Maximum Completion Time) also given us inputs. New priority rule is created based on the fuzzy rule. Fuzzy rule always deals with the approxiamation of precise answer. The fuzzy logic is determined by the membership functions.

$$A = \{(y, \mu_A(y)) | y \in U\} \quad \text{.....(1)}$$

This equation (1) clearly denotes the membership function of the fuzzy logic .

Were

$\mu_a(.)$ denotes membership function of A

U denotes Universal set of ordered pairs

A is the fuzzy set

y denotes the elements in the set A.

Fuzzy rule inference rule is used to generate new priority rule.Mandomi rule is used to select user priority tasks with larger task length.

Table 3.0 Fuzzy Inference Rule

Inputs		Output
UP	MCT	NP
Very Low	Low	Very Low
Very Low	Medium	Very Low
Very Low	High	Low
Low	Low	Very Low
Low	Medium	Medium
Low	High	Medium
Medium	Low	Low
Medium	Medium	Medium
Medium	High	High
High	Low	Medium
High	Medium	High
High	High	Very High
Very High	Low	Medium
Very High	Medium	High
Very High	High	Very High

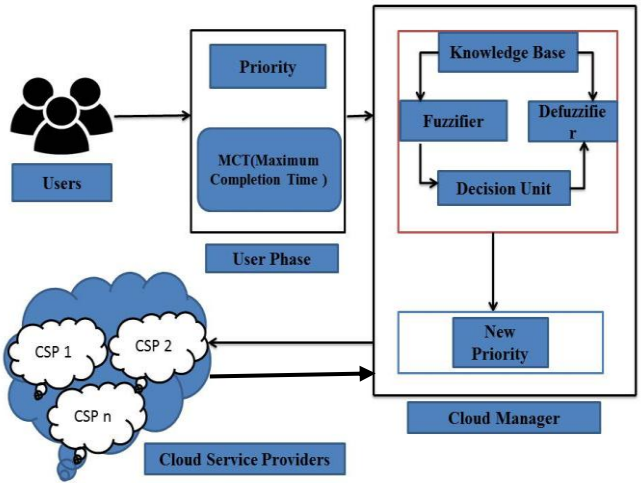

Fig 3.0 Priority based Multi Cloud Cloud Model

The fig 3.0 clearly depicts the working priority based multicloud model. The model comprised of three main features namely cloud users, cloud manager and Cloud service providers. Cloud users are the user input phase , in this phase the user gives input tasks with the priorities. The task length are also predefined to the user prioritized tasks. Cloud Manager acts as broker between service producer and consumer. Cloud manager enables fuzzy rule based on the user inputs and tasks with larger length. Cloud manager generates new priority based on the Mandomi Defuzzification rule. Fuzzification is used to Compare the input variables with the membership functions on the antecedent part to obtain the membership values of each linguistic label. The centroid Defuzzification method is used to get crisp output values. The center of gravity method is used to get output and it is formulated as

$$x^* = \frac{\int \mu_A(x).x\,dx}{\int \mu_A(x)dx} \quad \textbf{.....(2)}$$

The Fuzzifier converts the crisp input values into lingusistic values. Knowledge base is used for the decision making . IF THEN rule is used for taking decisions. The Fuzzy inference rules are stated in the table 3.0.

The new priority is computed by using Mandami Fuzzy Inference system. The inputs are measured by using linguistic variables. User set priority is categorized as Very High, High, Medium, Low,Very Low and Maximum Completion time are characterized as High,Medium and Low. The proposed rule is merely emphasized on the tasks with larger length.

© 2018, IJCSE All Rights Reserved

LAHUBMAX Algorithm

Pseudo code of LA HUB Max Algorithm

Input: Meta tasks MTa ,Virtual Machines Resources VMr, User Priority (UP), Maximum Completion time (MCTi)

Output : Mapped Schedule of Virtual Machine resources VMr with tasks MTa $S(Mt_a(RS_i))$

1. Begin
2. Initialization
   $MT_i \leftarrow \{ Ta_1 , Ta_2 ,,,,,,Ta_n \}$ // Meta tasks //
   Vmr $\leftarrow \{ Vma_1 , Vma_2 , ,,, Vma_m \}$ // Resources//
   Rtj $\leftarrow$ 0 // Ready time //
3. Read ETC Matrix ETCij // Expected Execution Time //
4. Repeat
   Until
   $MT_a \neq \emptyset$ // Meta tasks not empty //
5. For all tasks $Ta_i \in$ MTa
6. for all resource VMa $\in$ VMr
7. Compute Completion Time
   CTab = ETCij + Rtj
8. End for
9. End for
10. Do
    Until
    All tasks MT_i are mapped to the Virtual Machine Resources Vmr
11. For each Task Tai
12. find the minimum earliest completion and the resources that provide VMrj
13. Find the Task Tai with the maximium earliest completion time
14. Assign Tasks to VMrj based on the New priority Fuzzy Rule
15. Assign Tasks Tai to the resource Vmrj that gives earliest completion time
    MTa = MTa – Tai // delete the tasks from the list //
16. Update the Ready time Rtj = RTj + ETCij
17. Compute Mkspa = max (CT (MTi(VMrj)))
18. End for
19. End For
20. End

In the pseudo code of the proposed LAHUBMAX algorithm , let us consider MTai be the group of meta tasks and Tai be the individual tasks requested by the user to the server. VMr be the group of Virual machine servers, were VMa be the individual virtual server dedicated to complete the user requested tasks. The ETC Matrix (Expected execution time) matrix table is calculated by size of the task (individual processing capability of each task) and the Overall Capacity of the Virtual machines. Task length is termed MI (Instruction Volume) and Total capacity of the Virtual machine is termed as MIPS (Million Instruction per second.

The Million Instruction denoted as a task length and it is termed as (MI)

MI = SZ(Tai) ……. (3)

where SZ is the size of the of the individual processing element of the task The MIPS be the Millions Instruction per second is the overall capacity of the Virtual machine

MIPS = ∑ Vmr …….. (4)

where VMr stands for the Virtual machine capacity The ETC Matrix is calculated by the formula by (3) and (4)

ETCij = MI / MIPS …… (5)

The proposed LAHUBMAX Algorithm computes the completion time of each task. The completion time of the task is calculated by the sum of ETC matrix value and the ready time of each task.

This strategy helps to load balance the tasks and improve the resource utilization rate of the server. The LAHUBMAX algorithm focuses on two objectives, and they are makespan and resource Utilization. The increased resource utilization helps the service providers to increase their profit and the users

*Resource Utilization:*In Addition to reducing the makespan, LAHUBMAX is effectively utilizing all resources available. Resource Utilization for single case is calculated by equation 4.

$$Tsr_i = \sum_{i=1}^{n} \frac{Ru_i}{Ra_i} \quad \text{…… (5)}$$

Where Tsr_i be the total number of Resource Utilization ratio

Ru_i be the Number of individual resources used

Ra_i be the total number of Resources available

The proposed LAHUBMAX algorithm used the resources effectively when it is compared with Min-Min and LBMM algorithm. Even though LBMM algorithm balances load effectively than Min-Min LAHUBMAX performs better than LBMM. The results demonstrated that the proposed algorithm outperformed traditional min-min and LBMM in terms of utilization of the idle resources.

IV. RESULTS AND DISCUSSIONS

The experiments are conducted in this section to prove that the proposed LAHUBMAX algorithm outperforms than the existing Max-Min algorithms. This paper, cloud sim is used as a simulation tool for modeling the applications in the algorithm. For the experimental results, this paper evaluates the performance of the algorithm with indexes such as makespan and average resource utilization rate. The classes of the cloud sim are extended to utilize the proposed LAHUBMAX algorithm. The number of tasks and Virtual machines are flexible to the users.

© 2018, IJCSE All Rights Reserved

The Table 4.1 clearly illustrates the ETC Matrix . Expected Time to Compute (ETC) is derived from the formula discussed in the section 3.

Table 4.0 ETC Matrix

C1	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
VM1	11	6	10	8	5	6	14	5	10	6
VM2	7	15	10	5	4	14	15	7	11	6
C2										
VM3	13	14	4	13	5	8	8	10	5	7
VM4	9	9	6	4	8	4	3	5	3	9

The Table 4.1 clearly tabulates the priorities given by the user.

Table 4.1 User Priority

Task	Priority
T1	Very High
T2	High
T3	Medium
T4	Low
T5	Very Low
T6	Very High
T7	High
T8	Medium
T9	Low
T10	Very Low

The Table 4.2 Clearly illustrates the executed Tasks of the LAHUBMAX algorithm.The results of this algorithm is derived from the table 3.0 and Table 4.1.

Table 4.2 LAHUBMAX Algorithm

C1	T2	T1	T6	T7	T3	T8	T4	T9	T10	T6
VM1	6					5				
VM2		7							6	
C2										
VM3								5		
VM4			4	3	8		4			4

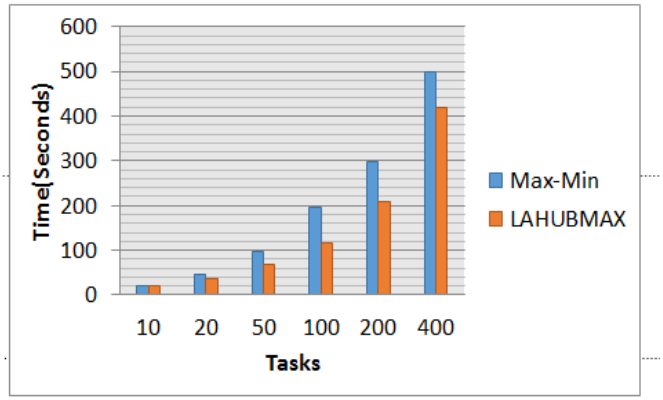

Fig 4.0 Performance Analysis(Makespan)

The fig 4.0 clearly depicts the performance analysis (makespan). The results show that the proposed algorithm outperforms the existing Max-min algorithm.

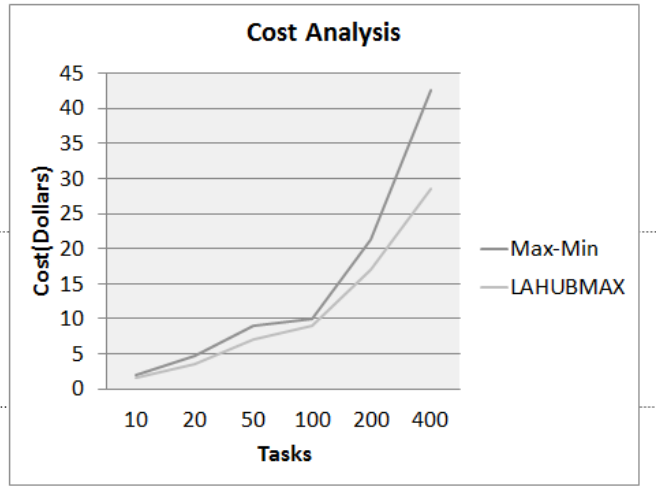

Fig 4.1 Cost Analysis

The fig 4.1 clearly illustrates the cost analysis of the proposed algorithm. The proposed algorithm reduces the execution cost of the virtual machines.

V. CONCLUSION AND FUTURE SCOPE

The proposed LAHUBMAX algorithm is used, to balance user requests that have larger number of tasks. It automatically load balances heavy weight resources with lighter weight resources and it also utilize the idle resources. User priority phase, is used to set the priority and to allocate and execute the tasks. The proposed algorithm outperforms the existing max-min in terms of makespan and cost. The future enhancement can be done in terms of meterics such as energy , fault tolerance and security. The proposed algorithm improves the cost around 30 % compared with the existing algorithm. Fuzzy logic is an novel mechanism is used in this

© 2018, IJCSE All Rights Reserved

algorithm to improve the performance of the multicloud systems.

REFERENCES

[1]. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, Hype, and Reality for Delivering computing as the 5th utility", Future Generation Computer Systems, ACM Volume25,No.6,pp:599-616,2009.

[2]. T Chatterjee, V. K. Ojha, M. Adhikari, S. Banerjee, U. Biswas and V. Snasel (2014), "Design and Implementation of a New Datacenter Broker policy to improve the QoS of a Cloud", Proceedings of ICBIA 2014, Advances in Intelligent Systems and Computing, vol. 303, pp- 281-290, 2014

[3]. BJ.Hubert Shanthan, L.Arockiam,"Resource Based Load Balanced Min Min Algorithm(RBLMM) for static Meta task Scheduling in Cloud",*International Conference on Advances in computer Science and Technology (IC-ACT'18) – 2018*, International Journal of Engineering and Techniques(IJET), Special Issue, pp.1-5, 2018.

[4]. V.A.Jane, B.J.Hubert Shanthan,. A Survey of Algorithms for Scheduling in the Cloud: In a metric Perspective, International Journal of Computer Sciences and Engineering,Vol.6,Special Issue 2, pp.66-70, 2018.

[5]. Botta, A., de Donato, W., Persico, V. and Pescape. A, "Integration of cloud computing and internet of things: a survey. Future Generation Computer Systems, Vol:56, pp.684-700, 2016.

[6]. Sirisha Potluri, Katta Subba Rao, "Quality of Service based Task Scheduling Algorithms in Cloud Computing", International Journal of Electrical and Computer Engineering (IJECE), Vol. 7, No. 2, pp. 1088-1095, 2017.

[7]. L. Benedict Carvin, BJ. Hubert Shanthan, A. DalvinVinothKumar, Dr.L.Arockiam, " Role of Scheduling and Load Balancing Algorithms in cloud to improve the Quality of Services", International Journal of Computer Science, Vol.5,No.1,pp:1454-1462, 2017.

[8]. BJ. Hubert Shanthan, A. DalvinVinothKumar, Er. Karthigai Priya Govindarajan , Dr.L.Arockiam, " Scheduling for Internet of Things Applications on Cloud: AReview",Imperial Journal of Inter disciplinaryResearch(IJIR), Vol.1, No.3, pp:1649-1653,2017.

[9]. Jiang, Hui, Jianjun Yi,Shaoli Chen, Xiaomin Zhu,"A multi-objective algorithm for task scheduling and resource allocation in cloud-based disassembly." Journal of Manufacturing Systems, Vol.41, pp:239-255, 2016.

[10]. Wang, G., Wang, Y., Liu, H., Guo, "HSIP: A Novel Task Scheduling Algorithm for Heterogeneous Computing.", Scientific Programming, pp:1-11, 2016.

[11]. Kamali Gupta , Vijay Katiyar , "Survey of Resource Provisioning Heuristics in Cloud and their Parameters", International Journal of Computational Intelligence Research, Vol.5,No.13,pp: 1283-1300, 2017.

[12]. B. Rajasekar, S.K. Manigandan, "An Efficient Resource Allocation Strategies in Cloud Computing", International Journal of Innovative Research in Computer and Communication Engineering, Vol. 3, pp.1239-1244, 2015.

[13]. Kokilavani T, Amalarethinam DD," Load balanced min-min algorithm for static meta-task scheduling in grid computing", International Journal of Computer Applications,Vol:20, No:2, :43-49. 2011.

[14]. Gaurang Patel, Rutuvik Mehta , Upenndra Bhoi , "Enhanced Load Balanced Min Min Algorithm for static meta task scheduling in cloud computing ",Elsveir Procedia Computer Science, ICRTC-2015, Vol:57,2015.

[15]. Maheswaran M, Ali S, Siegal HJ, Hensgen D, Freund RF. "Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems". In Heterogeneous Computing Workshop,.(HCW'99) Proceedings, IEEE, pp. 30-44, 1999.

[16]. Afaf Abd Elkader, "Enhancing the Minimum Average Scheduling Algorithm (MASA) based on Makespan Minimizing" , Artificial Intelligence and Machine Learning Journal, Vo. 17, No. 1, Delaware, USA, pp. 9-13, 2017.

[17]. Braun TD, Siegel HJ, Beck N, Bölöni LL, Maheswaran M, Reuther AI, Robertson JP, Theys MD, Yao B, Hensgen D, Freund RF. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. Journal of Parallel and Distributed computing.Vol:61, No.6, pp: 810-37. 2001.

[18]. Thomas A, Krishnalal G, Raj VJ. Credit based scheduling algorithm in cloud computing environment. Procedia Computer Science, Vol :31, No:46, pp: 913-20, 2015.

[19]. Parsa S, Entezari-Maleki R,"RASA: A new task scheduling algorithm in grid environment.", World Applied sciences journal, Vol:7, pp:152-60, 2009.

[20]. Sharma G, Banga P. "Task aware switcher scheduling for batch mode mapping in computational grid environment", International Journal of Advanced Research in Computer ScienceandSoftwareEngineering,Vol:3,No:6,pp:1292-1299, 2013.

© 2018, IJCSE All Rights Reserved