Utility-Driven Adaptive Scheduling for Cloud Service Provisioning

Manisha T. Tapale, R. H. Goudar, Mahantesh N. Birje

Abstract*: Though multiple cloud service providers (CSPs) offer similar services the access patterns of these services vary in general. Also, the CSPs belonging to different administrative domains will have different policies, preferences, and objectives for increasing their business profit as well as resource utilization. While Cloud user expects its service requests to be served as early as possible, CSPs try to serve user requests within deadline and obtain more profit (utility). Such utility motivated service provisioning may affect low utility jobs to starve from resources. Hence it is very much essential for CSPs to consider a deadline of jobs to avoid starvation. To address this issue we propose a utility-driven adaptive scheduling scheme that works as follows: 1) The service price, and thus the utility of service provisioning is determined using Non-cooperative bargaining protocol, 2) The scheduler prioritizes jobs depending on their utility and deadline, and then orders them in priority queue for execution. 3) On peak loads as new jobs with high utility arrive, some of the existing jobs having lesser utility in a priority queue start suffering from starvation. In such context, the proposed scheme adapts to generate another queue, called as FCFS queue, which holds relatively lesser profit jobs from priority queue. 4) The dispatcher then dispatches jobs to reliable virtual machines, choosing alternatively from priority queue and FCFS queue to achieve fair scheduling. The proposed scheme aims at maximization of the profit gained by CSP and avoidance of job starvation along with minimization of the service completion time and maximization of the utilization of resources. The proposed work is simulated using CloudSim and results show that it performs better than existing works.

***Index Terms*: Adaptive Scheduling, Cloud Computing, Priority, , Starvation, Service Provisioning, Utility.**

I. INTRODUCTION

Cloud computing represents a huge volume of computing, communication, and storage resources as well as various software resources for providing different services to organizations and public. It has become the most essential technology in business, education, research community and other sectors as many cloud services are coming up day-by-day. Cloud computing provides everything as a service to the users – Software as a service (SaaS), platform as a service (PaaS), and Infrastructure as a service (IaaS). Each service in a cloud computing is based on the pay-per-use of the resources [1].

In any type of service that the cloud provides - SaaS, PaaS, IaaS - the user needs to submit the service request (job) to the cloud service provider. The CSP then uses a scheduling mechanism to allocate the jobs to various resources in its pool. Cloud business environment exhibits dynamics in the market [4] in terms of resource supply, resource demand, service pricing, criticality of service executions, etc. the scheduler is continuously affected by such dynamics and heterogeneity in cloud environment. Also the scheduler has to schedule jobs considering different requirements mentioned by the user, and needs to consider number of constraints - type of the task, the size of the task, the task execution time, the availability of resources, the task queue, and the load on the resources. Thus scheduling of user jobs to resources becomes complex activity in the cloud.

There are many cloud service providers (CSPs) providing similar services, thus forcing them to provide best services with competitive costs to users. Also, Service level agreement policy, pricing of services, Service access control etc. will vary as they belong to different administrative domains. However, CSPs would like to offer services at best possible higher price and also optimize the utilization of their resources. On the other hand, cloud users would like to obtain the services at best possible lower price within a specified time limit. Thus it leads to conflicting interests of both stakeholders of cloud - the CSP and the user [2, 3]. Hence each one wishes to negotiate on the prices of the cloud services and tries to optimize his/her utility. But, such utility motivated service provisioning may affect lesser utility jobs to starve from resources. Hence it is essential for CSPs to consider a deadline of jobs to avoid starvation.

From above discussions we observe that there is a need of some new job scheduling scheme in the cloud environment that aims at adapting as per market dynamics, and provides economical and cost competitive services, in addition of minimizing makespan and optimizing resource utilization. Hence this work proposed a Utility-driven adaptive scheduling scheme for cloud service provisioning. The scheme works as follows: 1) The service price, and thus the utility of service provisioning is determined using Non-cooperative bargaining protocol, 2) The

Revised Manuscript Received on June 7, 2019

Manisha T. Tapale, Department of Computer Science and Engineering, KLE Dr. MSSCET, Belagavi, India.

R. H. Goudar, Center for Post Graduation Studies, Visvesvaraya Technological University, Belagavi, India.

Mahantesh N. Birje, Center for Post Graduation Studies, Visvesvaraya Technological University, Belagavi, India.

scheduler prioritizes jobs depending on their utility and deadline, and then orders them in priority queue for execution. 3) On peak loads as new jobs with high utility arrive, some of the existing jobs having lesser utility in a priority queue start suffering from starvation. In such context, the proposed scheme adapts to generate another queue, called as FCFS queue, which holds relatively lesser profit jobs from priority queue. 4) The dispatcher then dispatches jobs to reliable virtual machines (VMs), choosing alternatively from priority queue and FCFS queue to achieve fair scheduling.

The major contributions of this paper are summarized as follows.

1. Determination of the utility of service provisioning by applying game theory based non-cooperative bargaining protocol.
2. Prioritization and ordering of cloud jobs depending on their utility and deadline
3. Dynamic adaptation of scheduler to generate a multilevel queue on peak loads.
4. Dispatcher avoids starvation of jobs and achieves fair scheduling through switching between these queues
5. Assurance of job execution by mapping them to reliable virtual machines

The paper is organized as follows: Next Section presents literature survey which focuses on the literature study of scheduling mechanisms in cloud. The 3rd section on proposed work describes the proposed system in detail along with algorithms for adaptive scheduling scheme, job prioritizing, adaptive queuing mechanism and computing reliability of VM's. Section 4 describes about Simulation and Results of the proposed scheme. Finally, concluding remarks are presented in Section 5.

II. LITERATURE SURVEY

This section provides review of various works on resource pricing, resource brokering, and scheduling. A survey of cloud concepts, challenges and resource provisioning is given in [1, 5], which discuss the current status of resource provisioning methods and future directions. The work in [6] presents taxonomy of task scheduling in distributed cloud computing. Various works on scheduling and allocation of resources are given in [7, 8, 9, 10]. The work in [9] discusses scheduling of jobs and dispatching strategy. A comparative analysis of task scheduling approaches is given in [10]. Genetic and evolutionary algorithm based scheduling is discussed in [11, 12, 13, 14]. Papers [15, 16] discuss about scheduling to minimize the makespan of jobs.

There exists some priority based scheduling mechanisms [17, 18, 19, 20, 21, 22]. These algorithms apply methods like analytical hierarchy process, weighted fair scheduling, and round robin algorithm to prioritize the tasks and schedule. Various works are available in the literature which tries to consider QoS parameters while scheduling [23, 24]. Some works focus on optimizing the parameters like energy, bandwidth, cost etc. Papers in [25, 26] present works on energy aware task scheduling. The paper [27] presents work on bandwidth aware task scheduling, while [28] focuses on deadline based resource scheduling. Trust based scheduling is mentioned in [29].

A survey on SLA brokering is given in [30]. Some works are available [31, 32, 33] on taxonomy and brokering of interconnect and inter cloud architectures and discuss about interoperability and portability issues. Some of the factors considered for interoperability adoption are geographical distribution, scalability, vendor lock-in , and reliability of cloud resources.

Many works exists on resource pricing using different mechanisms. The paper [34] focuses on auction based resource provisioning. The works given in [35 - 39] discuss on pricing issues related to cloud resources while scheduling. The work in [3, 40, 41, 42] discuss about computation of resource price based on game theory concept to satisfy the needs of CSP users. They considered preferences of both users and CSP, requirements of users, and market dynamics while calculating the resource price. The paper [43] presents a work on scheduling of jobs considering their priority based on cost and deadline.

From the above discussions, we observe that though there exist many works on scheduling in cloud computing, they schedule task to any available resource as soon as it arrives without considering the cost of a task. This may lead to a problem of over-pricing of cloud services when many simple tasks are there, and/or under-pricing when some complex tasks are there for execution in the cloud. Also, some of the jobs might starve for resources. The proposed scheme is discussed in the next section to address this issue.

III. PROPOSED SCHEME

The main aim of cloud computing is to provide optimal scheduling of jobs and provide the users and CSP a profit, with improved QoS in minimum execution time, and at the same time balance the load. Job scheduling in this context could mean to provide an optimal mapping of jobs to appropriate resources. A CSP tries to obtain more profit while mapping jobs to reliable resources. Figure 1 shows a block diagram of the cloud scenario which is composed of users, resource broker, CSP, and virtual machines.

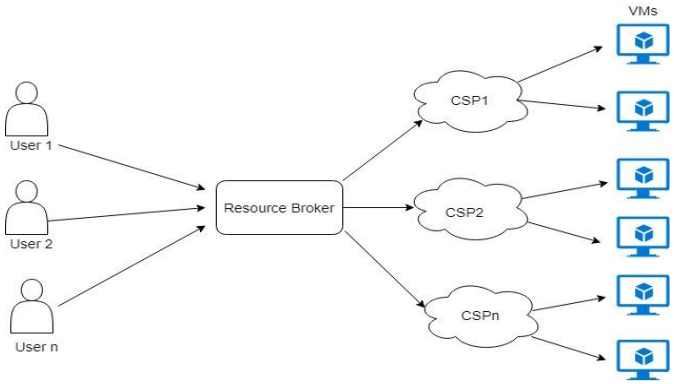

Fig. 1: Block Diagram of Cloud Environment

In a cloud environment user submits a service request (job) to the CSP using front end interface. It considers an intermediate entity called resource broker, to negotiate service price on behalf of users. An overview of the proposed utility driven adaptive scheduling scheme adopted by Scheduler of CSP is given in Algorithm 1.

Algorithm 1: Utility Driven Adaptive Scheduling Scheme

Input: Job set J = $\{j_1, j_2, ...j_n\}$ from different users U = $\{u_1, u_2, ...u_m\}$ at time interval T = $\{t_1, t_2, ...t_p\}$

Output: Priority based scheduling of jobs to different VM's
VM = $\{vm_1, vm_2, ...vm_o\}$ considering their reliability

Initialize: maximum Queue size Q_{max} such that Q_{max}>n where n is the number of jobs in J at time t_i th as the threshold of the queue.

1. User submits the service request (job) to CSP for execution
2. Service price is negotiated by the Resource broker using (non-cooperative bargaining game) and the utility is determined
3. Local scheduler will prioritize and order the jobs based on utility and deadline using Job Prioritizing algorithm
4. On peak loads Local scheduler adapts dynamically to generate a multilevel queue (two queues called Priority Queue and FCFS Queue)
5. VM Manager calculates the reliability of each VM and maintains VM status information
6. Dispatcher in interaction with VM Manager identifies reliable VMs, and schedules jobs to them, choosing jobs from multilevel queue such that jobs starvation is avoided.

The cloud users U submit a set of jobs J to the CSP. The resource broker, on behalf of user, bargains its price and utility(surplus) value is determined by playing a bargaining game with different service providers [2]. Local scheduler dynamically adapts to create multilevel queues and orders jobs as per heir priorities. VM manager calculates reliability of VMs. Dispatcher maps jobs from multilevel queues to reliable VMs for execution. These steps are elaborated further in the following subsections:

A. Service Price Negotiation

The users use an interface to submit their jobs. The further interaction between users and CSPs takes place via resource broker to negotiate price and execute jobs. The intermediate resource broker assists cloud users to obtain services from different CSPs. The resource broker applies a non-cooperative bargaining protocol based pricing strategy to decide upon a service cost. Algorithm 2 describes the process of negotiation.

Algorithm 2: Price Negotiation using a Non-Cooperative Bargaining Protocol

Begin

1. User job requests arrive to resource broker with their requirements like deadline and price.
2. Initialize the reserved prices of users (Rb) and CSPs (Rp)
3. Calculate the initial offered price of resource broker (Ob), and CSP (Op) as Ob = 50 % of Rb, and Op = 150% of Rp
4. Compute the next offered prices Ob [], and Op [] of resource broker, and CSP respectively; And also, initialize the probability of acceptance of these prices Pb (Ob[ ]) , and Pp (Op[ ]) of resource broker, and CSP respectively for broker (Ob[ ], Pb (Ob[ ])) and CSP (Op[ ], Pp (Op[ ])).

// Let CSP initiates the bargaining game

5. CSP offers his offered price, Op

//loop begins

6. While ((Price not accepted) and (not max. no. of negotiations))

// Resource Broker starts the bargaining process

7. If the Op <= Ob then
   Accept the price
   Else
   Compute the utility of broker
   Counter offer the price with maximum utility

// CSP continues the bargaining process

8. If the Ob >= Op then
   Accept the price
   Else
   Compute the utility of CSP
   Counter offer the price with maximum utility
9. Modify Pb (Ob[ ]) , and Pp (Op[ ]) of resource broker, and CSP by decreasing the probability of acceptance.

//loop ends

10. print the accepted price of resource or service

End

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

A detailed description of this protocol is given in one of our previous work [2]. Once the service price is negotiated between user and CSP, utility of CSP is calculated as the difference between the negotiated price and its reserved price.

B. Utility-Driven Adaptive Scheduling

A block diagram of the proposed Utility-driven adaptive scheduling scheme is as shown in Fig. 2. It elaborates various components of CSP shown in Figure 1. The CSP has three main components: Local scheduler, Dispatcher and Service Monitor, and a VM Manager. They are described as follows:

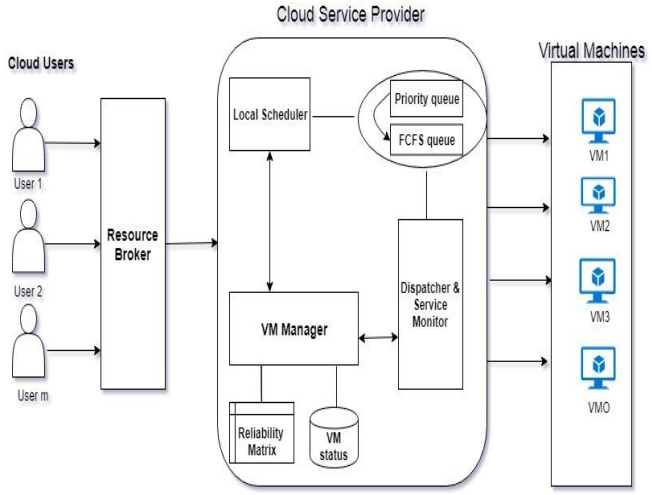

Fig. 2: Block Diagram of Utility-Driven Adaptive Scheduling

Local Scheduler

The local scheduler prioritizes and orders jobs for execution based on their utility and deadline. For a given job set Algorithm 3 describes the method of prioritizing and ordering of jobs.

Algorithm 3: Job Prioritizing Algorithm

Input: Job set J = { j_1, j_2.....j_n } from different users U ={ u_1, u_2, ...u_m }
Surplus S_{ji}
Deadline dl_{ji}

Output: Ordered (prioritized) job set J

1. for the job set J in a a given time interval T={t_1,t_2.....t_p}
2. for each job ji in the job set
3. Calculate the weightage W = { w_{j1},w_{j2}, w_{j3}.....w_{jn} } as W_{ji} = 0.6 S_{ji} × 0.4 dl_{ji}
4. end for
5. Order the jobs in a decreasing order of their weightage.

Job prioritizing algorithm takes a job set as an input from different users, along with the profit (*utility*) S_{ji} for each job, and the *deadline* dl_{ji} in which it should complete the job. The utility and deadline are given a weightage of 60% and 40% respectively. After all the weightages are calculated for the jobs in a given time interval, the jobs are ordered in decreasing values of their weightage in the priority queue for execution.

For newly arriving jobs and/ or during peak loads in a cloud, there is a chance that most of the new jobs may provide a high surplus compared to the ones' already in the priority queue. The existing jobs in a priority queue may starve due to these new jobs. To avoid such starvation, scheduler adapts to generate multilevel queue. Algorithm 4 describes an Adaptive Queuing Mechanism to avoid starvation of jobs.

When you submit your final version, after your paper has been accepted, prepare it in two-column format, including figures and tables.

Algorithm 4: Adaptive Queuing Mechanism for Starvation Avoidance at Scheduler

Input: Ordered job set J
Output: Multilevel queue generation
Initialize: Q_{max} such that $Q_{max} > n$
Create a priority queue using a doubly linked list(DLL) with a single node

1. for each job j_i in the ordered job set J
2. insert job j_i in the priority queue
3. if queue size > th
4. if FCFS does not exists
5. create and insert the last job in priority queue to FCFS queue
6. else insert the job of Priority queue into an FCFS queue
7. end for

The maximum size (Q_{max}) of the priority queue depends on the number of jobs (n). A threshold value *th* (80 percent of n) is considered. For every ordered job with a calculated weightage insert the job in the priority queue which is already ready as a part of the local scheduler for scheduling and check if the queue size has reached its threshold *th*. If a threshold is reached we create another level in the queue as an FCFS queue, and insert the last job in the priority queue into the FCFS queue.

VM Manager

After the jobs are mapped to resources, the execution starts affecting the status of resources (VM). Thus monitoring of the status and health of resources is essential [44, 45]. The VM manager is responsible for monitoring the utilization (health) status of each VM at regular intervals. A database called VM STATUS stores all relevant status related parameters. The status of any VM gets affected by its load. As more number of

tasks are assigned to a VM its utilization increases, which may also have negative impact on its reliability status. Hence maintaining information of VM's reliability is also important. So a VM manager calculates and maintains the reliability status of VMs.

The reliability of a VM *i*, *Rvmi* , can be defined as its ability to perform its functions in all circumstances, as well as any unexpected circumstances. We consider the parameters like power consumption, availability of memory, bandwidth and expected job completion time to compute the reliability of a virtual machine. The equation 1 shows the calculation of VM's reliability.

$$Rvmi = \alpha 1 \frac{dl_{ji}}{J_{ECT}} + \alpha 2 \frac{POW_{avail}}{POW_{req}} + \alpha 3 \frac{M_{avail}}{M_{req}} + \alpha 4 \frac{B_{avail}}{B_{req}} \quad (1)$$

where

- dl_{ji} is the deadline of the job given by the user.
- J_{ECT} is the expected completion time of the job.
- POW_{avail}, M_{avail} and B_{avail} is the current availability of battery power, memory and bandwidth
- The required power, memory and bandwidth is represented as POW_{req}, M_{req} and $B_{req.}$
- α1, α2, α3 and α4 are the weights associated with each parameter and their importance in job scheduling. The weights are assigned as $\sum_{i=1}^{4}(\alpha i = 1)$. Proposed scheme considers deadline along with the utility to prioritize jobs. Hence we assign relatively more weightage to α_1 in this equation.

The detailed description about computation of J_{ECT}, POW_{req}, M_{req} and $B_{req.}$ is provided in our previous paper [5].

Algorithm 5 describes the method of calculating reliability of VMs.

Algorithm 5: Computing Reliability of VM's at VM Manager

Input: VM = { vm_1, vm_2, vm_3.....vm_o}

Output: Reliability matrix of VMs

1. Identify a set of VMs
2. Monitor the values of Bandwidth, Power, Memory of each VM.
3. For each VM calculate the reliability of each vm as follows
4. $Rvmi = \alpha 1 \frac{dl_{ji}}{J_{ECT}} + \alpha 2 \frac{POW_{avail}}{POW_{req}} + \alpha 3 \frac{M_{avail}}{M_{req}} + \alpha 4 \frac{B_{avail}}{B_{req}}$
5. end for
6. Sort the VMs in descending order of their reliability
7. Update the reliability status of VMs.

Dispatcher and Service Monitor

The dispatcher interacts with the VM manager to get the reliability information of the VMs. Then it dispatches the jobs to relevant reliable VMs from priority queue. If jobs are available in FCFS queue also, then the dispatcher switches alternatively between priority queue and FCFS queue for dispatching of jobs. This approach gives equal chance of execution of the jobs available in priority queue (relatively high utility) as well as those available in FCFS queue (relatively lesser utility). Thus fair scheduling is achieved and starvation of lesser profit jobs is avoided.

IV. SIMULATION AND RESULTS

CloudSim is used to simulate the functioning of utility-driven adaptive scheduling scheme. The system configuration during the simulation of proposed scheme was Intel (R) Core (TM) 2 Duo CPU processor T6670 @ 2.20 GHz and 2 GB RAM running Windows XP 7. Table 1 represents various parameters used in simulation of the proposed work.

Table I : Simulation Parameters

Parameter Name	Value
Number of Physical Machines	10 - 20
Number of Virtual Machines	50 - 100
Number of Customers	10 - 20
Number of Metatasks (jobs)	50 - 200
Number of Tasks per Metatask	20 - 100
Number of Cloudlets	1500
Size of metatask	100 - 500 MI
Job arrival rate	50 / unit time
Processing Speed	50 - 300 MIPS
Bandwidth of Links	100 - 200 Mbps
Cost of processor	0.2 - 0.5 $ / unit time
Cost of bandwidth	0.01 - 0.03 $ / unit time
Service / processing rate of a VM	5 - 10 jobs / sec
Threshold (th)(Maximum allowed utilization of VM)	0.7

Various performance metrics considered are as follows:

Priority: It is calculated for each job based on its deadline and profit that it returns for provisioning of the service.

Utility: it is the profit obtained by CSP using a non-cooperative bargaining game after negotiating for service provision. It is calculated as the difference between negotiated price and reserved price.

Reliability: The reliability of a VM is defined as its ability to perform its functions in all circumstances such as under load, peak load, as well as any other unexpected situation. It is calculated using task deadline, power, bandwidth and memory requirements.

Job Completion Time (Makespan): it is the total time taken to play concurrent bargaining Game, prioritize and order jobs in a multilevel queue, and dispatch jobs and execute on reliable VMs.

Resource Utilization Rate : it represents an amount of load on the resource or a VM. It is defined as the ratio of job arrival rate to the processing rate of a VM.

Results

Some of the simulation results obtained are discussed below.

Reliability: Job deadline of the task is given more weightage as the scheme prioritizes the task based on its utility and deadline. Fig. 3 shows the reliability values in percentage for 100 VMs at particular time. The scheme considers a VM reliable for scheduling if its reliability value is greater than or equal to 30%. from the figure 3 we observe that reliability range for most of the VMs falls between 30% - 60 %. As the scheme prefers more reliable devices for scheduling and successful execution of jobs, their resource utilization rate will be increased; thus reducing slowly the reliability value of such reliable VMs over the time.

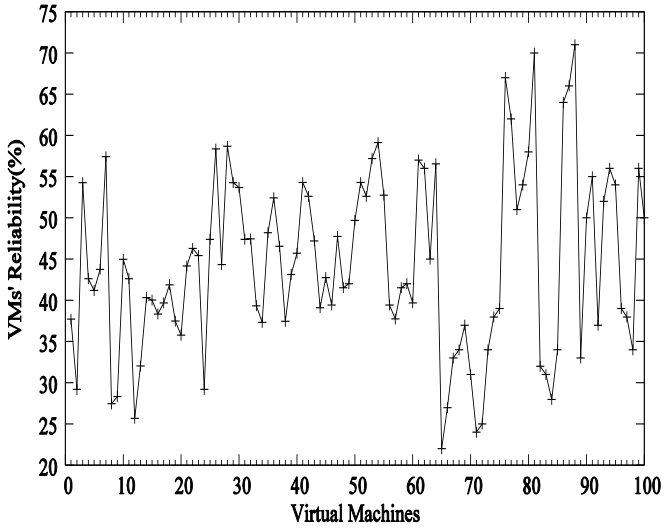

Fig. 3: VMs' Reliability Versus Virtual Machines.

Resource Utilization: Fig. 4 shows resource utilization of the proposed work compared to the scheme given in [43]. It is observed that resource utilization of the VMs in proposed work is mostly in the range 30-60; after scheduling tasks to most reliable VMs, their resource utilization increases, reducing slowly their reliability; thus the scheme selects other VM's while mapping the next tasks, distributing the load across all VMs. Where as in [43], the resource utilization of VMs is random because it does not consider the reliability of the VMs while scheduling. Thus, this leads to over utilization of some VMs and/or under utilization of other VMs.

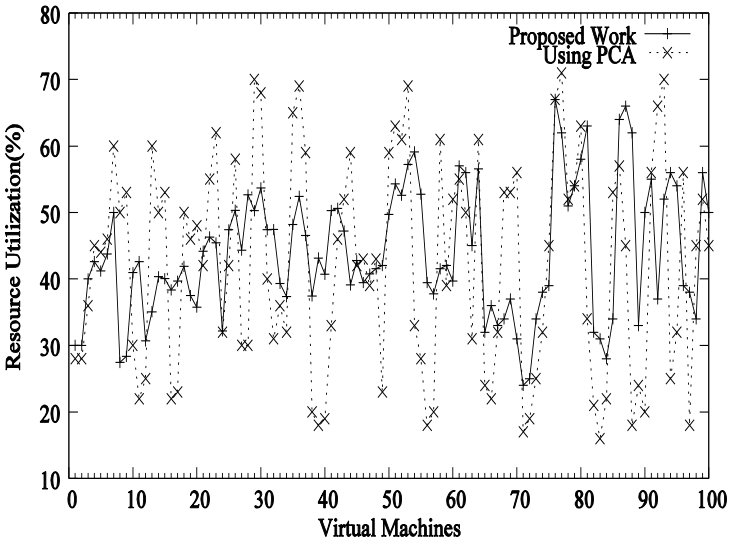

Fig. 4: Virtual machines Versus Resource Utilization

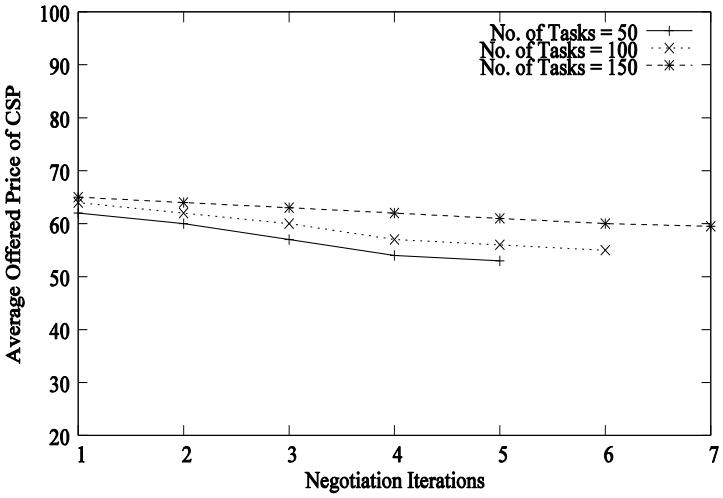

Fig. 5: Negotiation Iterations Versus Average Offered Price of CSP

Average Offered price for CSP: Fig. 5 represents the average offered price at different negotiation iterations for different number of tasks. Initial offered price varies depending on the number of tasks. It is observed that as the number of iterations increase the offered price is reduced to converge the bargaining game. Also observed that game converges faster when number of tasks is less, and it takes more negotiation iterations for more number of tasks.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

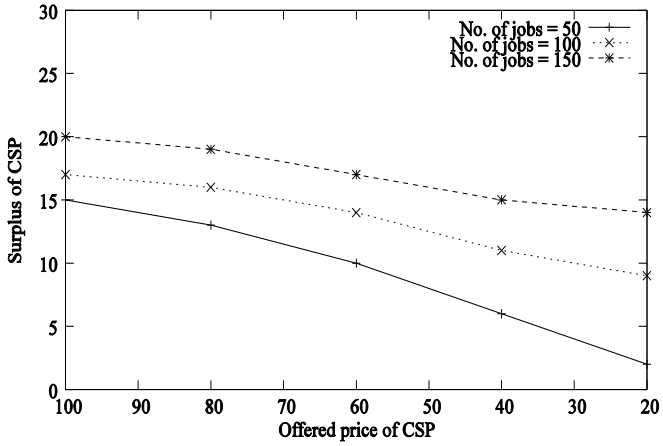

Fig. 6: Offered price of CSP Versus Surplus of CSP

Utility (Surplus): Fig. 6 represents an average surplus (utility) of CSP for different number of tasks obtained after negotiation. It is observed that the surplus decrease with offered price at different negotiation iterations. Also when number of tasks is more surplus obtained is high. This is due to more load on VMs during more number of jobs, and the charges increase along with the load.

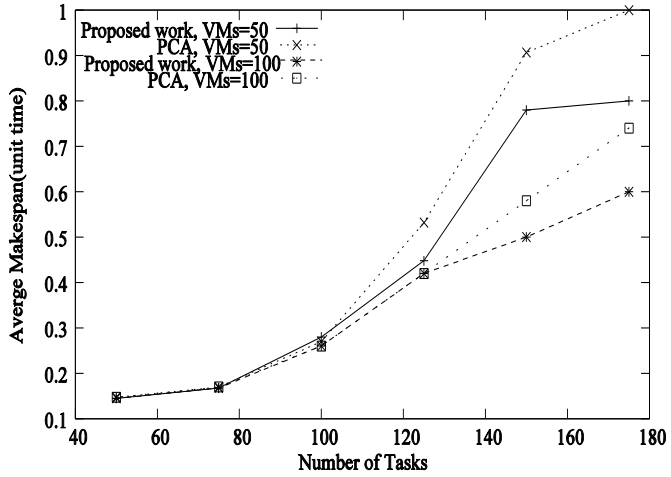

Fig. 7: Number of Tasks Versus Average Makespan (unit time)

Average Makespan: Fig. 7 represents an average makespan versus number of tasks considering different number of VMs. Makespan increases slowly up to some number of tasks, but increases rapidly with further increase in number of tasks. Average makespan matches with that of [43] when number of jobs are less, but it is lesser in the proposed work compared to [43] after further increase in the number of jobs. In the proposed work the average execution time remains almost constant even after further increase in arrival of tasks at peak loads. This is because at peak loads the dispatcher in the proposed scheme switches between priority queue and FCFS queue to avoid starvation of any jobs. Hence proposed work has lesser makespan compared to [43], especially at peak loads.

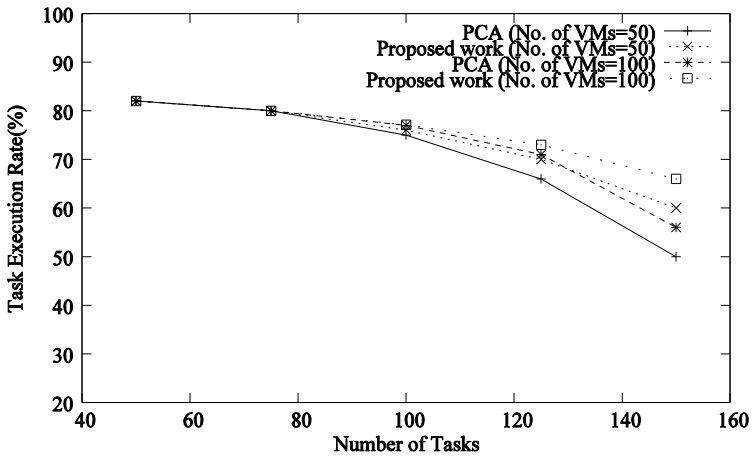

Fig. 8: Number of Tasks versus Task Execution Rate

Task Execution Rate: Figure 8 shows task execution rate versus number of tasks considering different number of VMs. The result shows that compared to [43], the execution of jobs is guaranteed more in proposed work because of mapping to more reliable VMs. As the tasks are mapped to reliable devices their execution is guaranteed, thus showing better performance over [43] even at peak loads.

V. CONCLUSION

A utility motivated cloud service provisioning may affect less profit jobs to starve from resources. Hence the proposed work on utility-driven adaptive scheduling considers a deadline of jobs along with profit to avoid starvation. It is used to determine the utility of service provisioning using non-cooperative bargaining protocol. Jobs are prioritized and ordered depending on their utility and deadline. Proposed scheduler adapts dynamically to generate a multilevel queue on peak loads and/or resource demand. The dispatcher avoids starvation of jobs and achieves fair scheduling through switching between priority queue and FCFS queue; and it also assures of jobs execution by mapping them to reliable virtual machines. Thus the proposed scheme helps in maximizing the profit of CSP and avoiding job starvation along with minimization of service completion time and maximization of resource utilization.

REFERENCES

1. Mahantesh N. Birje, Praveen Challagidad, R. H. Goudar, Manisha Tapale, "Cloud Computing Review: Concepts, Technology, Challenges and Security", International Journal of Cloud Computing (IJCC), Inderscience, Vol. 6, No. 1, pp. 32-57, 2017.
2. Goudar, R.H., Tapale, M.T., Birje, M.N., "Price negotiation for cloud resource provisioning", Proceedings of the 2017 International Conference On Smart Technology for Smart Nation, SmartTechCon 2017, Bangalore, India.
3. M. N. Birje, S. S. Manvi, Chetan Bulla, "Economical Job Scheduling in Wireless Grids", Third Int. conference on Electronics Computer Technology, ICECT 2011, Kanyakumari, India.

4. Mahantesh N. Birje , Sunilkumar S. Manvi , Sajal K. Das, "Reliable resources brokering scheme in wireless grids based on non-cooperative bargaining game", Journal of Network and Computer Applications, 39, p.266-279, March, 2014 [doi:10.1016/j.jnca.2013.07.007]
5. Singh S, Chana I. "Cloud resource provisioning: survey, status and future research directions". Knowledge Information System, 49(3), 2016.
6. Casavant, T., Kuhl, J. G., "A Taxonomy of Scheduling in General-purpose Distributed Computing Systems", IEEE Transactions on Software Engineering, Vol.14, No. 2, 141-154,1988.
7. Ergu D, Kou G, Peng Y, Shi Y, Shi Y, "The analytic hierarchy process: task scheduling and resource allocation in cloud computing environment", The Journal of Supercomputing. 64(3):835-848, 2013.
8. Maguluri ST, Srikant R, "Scheduling jobs with unknown duration in clouds", IEEE/ACM Transaction on Networking (TON), 22(6):1938–1951, 2014.
9. Tai-Lung Chen et al, "Scheduling of Job Combination and Dispatching Strategy for Grid and Cloud System", GPC 2010, 612–621, 2010.
10. A. Jain and R. Kumar, "A Comparative Analysis of Task Scheduling Approaches for Cloud Environment", International Conference On Computing for Sustainable Global Development, pp. 2602-2607, 2016.
11. Tsai J-T, Fang J-C, Chou J-H, "Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm", Journal on Computers & Operations Research, Vol. 40(12), pp. 3045-3055, 2013.
12. Liu X, Zha Y, Yin Q, Peng Y, Qin L, "Scheduling parallel jobs with tentative runs and consolidation in the cloud", Journal of System Software, 104:141–151, 2015.
13. A. Tumanov, J. Cipar, G.R. Ganger & M.A. Kozuch, "Algebraic scheduling of mixed workloads in heterogeneous clouds", In Proceedings of the Third ACM Symposium on Cloud Computing, 2012, pp. 25-30.
14. Singh S, Kalra M., 'Scheduling of independent tasks in cloud computing using modified genetic algorithm", Computational Intelligence and Communication N etworks (CICN), 2014.
15. Bhoi, U., Ramanuj, P.N. "Enhanced Max-Min Task Scheduling Algorithm in Cloud Computing", International Journal of Application or Innovation in Engineering & Management (IJAIEM), Vol. 2, No. 4, 259-264, 2013.
16. Liu, G., Li, J., Xu, J., "An Improved Min-Min Algorithm in Cloud Computing", Proceedings of the International Conference of Modern Computer Science and Applications. Springer, Wuhan, China, pp. 47-52, 2012.
17. Ghanbari, S., Othman, M., "A Priority-based Job Scheduling Algorithm in Cloud Computing", Proceedings of the International Conference on Advances Science and Contemporary Engineering (ICASCE). Jakarta, Indonesia, pp. 778-785, 2012.
18. Ghanbari S, Othman M, Bakar MRA, Leong WJ, "Priority-based divisible load scheduling using analytical hierarchy process", Applied Mathematics & Information Sciences, 9(5), pp. 25–41, 2015.
19. Li Yang et al, "A new Class of Priority-based Weighted Fair Scheduling Algorithm", Physics Procedia, 33, 942 – 948,2012.
20. Sunita Bansal et al, "Dynamic Task-Scheduling in Grid Computing Using Prioritized Round Robin Algorithm", IJCSI International Journal of Computer Science Issues, 8(2), pp. 472-477,2012.
21. Deepika Saxena, RK Chauhan, and Ramesh Kait, "Dynamic fair priority optimization task scheduling algorithm in cloud computing: Concepts and implementations", International Journal of Computer Network and Information Security, 8(2):41, 2016.
22. N. Jain, N. Grozev, J. Lakshmi and R. Buyya, "PriDynSim a Simulator for Dynamic Priority Based I/O Scheduling for Cloud Applications," 2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, 2015, pp. 8-15. doi: 10.1109/CCEM.2015.17.
23. Wu, X., Deng, M., Zhang, R, Zeng, B., Zhou, S., "A Task Scheduling Algorithm Based on QoS-Driven in Cloud Computing". Proceedings of the International Conference on Information Technology and Quantitative Management. Elsevier Procedia, Suzhou, China, pp. 1162-1169, 2013.
24. Tamilselvan L, Anbazhagi, Shakkeera, "Qos based dynamic task scheduling in Iaas cloud", IEEE International Conference on Recent trends in Information Technology (ICRTIT), 2014, pp. 1-8, 2014.
25. Cheng C, Li J, Wang Y, "An energy-saving task scheduling strategy based on vacation queuing theory in cloud computing", Tsinghua Science and Technology, 20(1), pp. 28–39, 2015.
26. Zhu X, Yang LT, Chen H, Wang J, Yin S, Liu X, "Real-time tasks oriented energy-aware scheduling in virtualized clouds", IEEE Transactions on Cloud Computing 2(2):168–180, 2014.
27. Lin W, Liang C, Wang JZ, Buyya R, "Bandwidth-aware divisible task scheduling for cloud computing", Software: Practice and Experience 44(2), pp. 163–174, 2014.
28. Rodriguez MA, Buyya R, "Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds", IEEE Transactions on Cloud Computing 2(2), pp. 222–235, 2014.
29. Wei Wang, "Cloud-DLS: Dynamic Trusted Scheduling for Cloud Computing", Expert Systems with Applications, 39, pp. 2321–2329, 2012.
30. E. Mostajeran, B.I. Ismail, M.F. Khalid, H. Ong, "A survey on sla-based brokering for inter-cloud computing", Second International Conference on Computing Technology and Information Management, ICCTIM, pp. 25–31, 2015.
31. A.N. Toosi, R.N. Calheiros, R. Buyya, "Interconnected cloud computing environments: Challenges, taxonomy, and survey", ACM Computing Surveys, 47 (1) pp. 7:1–7:47, 2014.
32. N. Grozev, R. Buyya, "Inter-cloud architectures and application brokering: taxonomy and survey", Software: Practice and Experiences, 44 (3), pp. 369–390, 2014.
33. S.S. Chauhan, et al., "Brokering in interconnected cloud computing environments: A survey", Journal on Parallel and Distributed Computing, 2018. https://doi.org/10.1016/j.jpdc.2018.08.001
34. S.A. Flor, F.L. Pires, B. Barán, "Auction-based resource provisioning in cloud computing. A taxonomy", Latin American Computing Conference, CLEI, pp. 1–11, 2015.
35. M. Aazam, E.N. Huh, M. St-Hilaire, C.H. Lung, I. Lambadaris, "Cloud customer historical record based resource pricing", IEEE Transaction on Parallel and Distributed Systems, 27 (7) pp. 1929–1940, 2016.
36. F. Fowley, C. Pahl, P. Jamshidi, D. Fang, X. Liu, "A classification and comparison framework for cloud service brokerage architectures", IEEE Transaction on Cloud Computing, PP (99) 1–1, 2017.
37. Z. Guan, T. Melodia, "The value of cooperation: Minimizing user costs in multi-broker mobile cloud computing networks", IEEE Transaction on Cloud Computing. PP (99),1–1, 2017.
38. Cao, Q., Wei. Z., Gong, W. M., "An Optimized Algorithm for Task Scheduling Based On Activity Based Costing in Cloud Computing", Proceedings of the 3rd International Conference Bioinformatics and Biomedical Engineering (ICBBE). IEEE Computer Society, Beijing, China, pp. 1-3, 2009.
39. Selvarani, S., Sadhasivam, G. S. "Improved Cost-based Algorithm for Task Scheduling in Cloud Computing", Proceedings of International Conference Computational Intelligence and Computing Research (ICCIC), IEEE Computer Society, Coimbatore, India, pp. 1-5, 2010.
40. Mahantesh N. Birje , Sunilkumar S. Manvi , Sajal K. Das, "Resource pricing strategy in wireless grid using non-cooperative bargaining game", 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, 2012. Pp. 61-66, India.
41. R. Buyya, C.S. Yeo, S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities", 10th IEEE International Conference on High Performance Computing and Communications, pp. 5–13, 2008.
42. R. Achar, P.S. Thilagam, "A broker based approach for cloud provider selection", International Conference on Advances in Computing, Communications and Informatics, ICACCI, 2014, pp. 1252–1257.
43. Naseem A.AL-Sammarraie et al, "A Scheduling Algorithm to Enhance the Performance and the Cost of Cloud Services" Computer Engineering and Intelligent Systems", Vol.6, No.8, 2015.
44. M. N. Birje, S. S. Manvi, "WiGriMMA: A Wireless Grid Monitoring Model using Agents", Journal of Grid Computing, SpringerLink, Vol. 9, no. 4, pp. 549-572, 2011.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

45. M. N. Birje, S. S. Manvi, "Monitoring and Status Representation of Devices in Wireless Grids", Grid and Pervasive Computing 2010 (GPC 2010) in Lecture Notes in Computer Science (LNCS), Taiwan: Springer Link, vol. 6104, pp. 341-352, May 2010.

AUTHORS PROFILE

Manisha T. Tapale, currently working as an Assistant Professor, Dept of CSE , KLE MSSCET, Belagavi. She has a 13 years of Teaching Experience. She has published papers in International Journals and Conferences. Her subjects of interest are Cloud Computing, Computer Networks, Network Security and Operating Systems.

R H Goudar, currently working as an Associate Professor, Dept. of CNE, Visvesvaraya Technological University, Belagavi.He has 14 years of Teaching Experience at Professional Institutes across India. He worked as a faculty at International Institute of Information Technology, Pune for 4 years and at Indian National Satellite Master Control Facility, Hassan, India. He published over 130 papers in International Journals, Book Chapters and Conferences of High Repute. His Subjects of Interest include Semantic Web, Network Security and Wireless Sensor Networks.

Mahantesh N. Birje received B.E. and M.Tech. degrees in Computer Science and Engineering in 1997 and 2005 respectively. He obtained Ph.D. from Visvesvaraya Technological University (VTU), Belagavi, India in 2012. His current research areas include Cloud Computing, Internet of Things, Data Mining, and Security. He has published many research papers in International refereed journals and Conferences. He is a Reviewer of some International Journals of IEEE, Elsevier, Springer, etc. He has given few invited Lectures and has organized Workshops and Seminars for Faculty and Students. He has executed various academic and administrative responsibilities. Currently he is working as Professor in the Center for Post Graduate Studies, VTU, Belagavi.