Deadline Constrained based Resource Allocation in Cloud Environment

N. Malarvizhi, Aswini. J, T. Kumanan

Abstract: Cloud computing faces a challenge of handling huge amounts of data. The users keep on pushing the data without knowing the challenge in increased storage. Task Scheduling deals with allocating the task to a respective resource pool on a demand basis. Approaches have been built that handle requests from users with deadlines on the amount of request that can be handled. It is important to understand that the mechanism is available to handle the deadlines. The experimental results show that the proposed algorithm produces remarkable performance improvement rate on the total execution cost and total transfer time under meeting the deadline constraint. In view of the experimental results, the proposed algorithm provides a better-quality scheduling solution that is suitable for scientific application task execution in the cloud computing environment.

Keywords: Task scheduling, Deadline constraints, Resource Allocation, Cost optimization.

I. INTRODUCTION

In cloud computing, a pool of resources is offered to the on-demand users on the basis of "pay per use". The resources are virtualized and allocated to the users. Different cloud models are adopted with different task scheduling or VM allocation mechanisms (Patidar et al 2012). Since, different user applications may require different on-demand resources, allocating and measuring the performance of the system is a challenging one in cloud computing. The challenges of resource utilization, direct us towards better task scheduling algorithms without violating SLA (Service Level Agreement). In cloud computing, each data center consists of multiple physical machines. In a single physical machine, different Virtual Machines (VM) are created as per the user request for computing application. Moreover, the user requests may have different parameter constants such as start time, execution time, number of nodes (Number of VMs) and deadlines (Abaker et al 2015).

In an instance, cloud computing processes hundreds or more than thousands of different applications and allocate on-demand resources.

Revised Manuscript Received on October 30, 2019.
* Correspondence Author

N. Malarvizhi*, Professor, Department of Computer Science and Engineering, School of Computing, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai-600062, Tamil Nadu, India. E-mail:drnmalarvizhi@gmail.com

Aswini. J, Research Scholar, Department of Computer Science and Engineering, Meenakshi Academy of Higher Education and Research, Chennai – 600 069, Tamil Nadu, India. & Assistant Professor, Department of Computer Science and Engineering, Saveetha School of Engineering, Chennai - 602 105, TamilNadu, India.E-mail:aswini.jayaraman@gmail.com

T. Kumanan, Professor, Department of Computer Science and Engineering, Meenakshi Academy of Higher Education and Research, Chennai – 600 069, Tamil Nadu, India. E-mail:kumananvetri@yahoo.com

The resource management mechanism allocates and releases the resources from one application to another (Ruiz-Alvarez et al(2015). Since the physical resources are limited, a cloud service provider implements optimized resource mechanisms to all its customers. However, the physical resources are limited in the capacity of CPU, memory, storage space, I/O devices, bandwidth and much more for the data center. The exact amount of resources should be provided to the user on-demand, which avoids over-utilization and under-utilization (Kumar Mohit & Sharma 2018). The idea of resource management is to utilize the resource as much as possible. The resource utilization can be achieved by implementing better task scheduling in cloud computing (M. Xu et al(2018).

II. LITERATURE SURVEY

The execution of a map task consists of reading, map, collect, spill, and merge phases. The map task reads and processes its corresponding data split. The intermediate outputs are partitioned to be further processed by reduce tasks. The execution of a reduce task includes shuffle, merge, reduce, and write phases (C. Chen et al (2018)). The reduce task fetches and sorts the intermediate data from map tasks. Finally, each reduces task processes the data and writes the output to the distributed file system. The inter-task dependency between map and reduce tasks is shuffle phase that transfers map outputs to reduce tasks and the shuffle is actually a part of the reducing phase. By reducing to two subparts, job execution within deadline is achieved (Liu & Buyya 2018).

As the name suggests this system comes is aware of cost. The Cost - Aware ability is because of the estimation of the configuration in such a way that the cost of executing the jobs is minimal. This depends on various types of servers. However for every type of server there is a billing cost and cost interval associated with it. It executes a special type of algorithm to estimate cost. The model makes use of a limited set of server instances I, based on a set of server types S(H. Moens et al (2013)). The set I must be chosen in such a way that there are enough servers to lead to a feasible flow, but not too many servers ensuring the algorithm execution does not become too complex**.**

A Directed Acyclic Graph G = (T,E,L) is used to model a data-intensive application. T={$\tau 1,\tau 2,...,\tau n$} is a set of nodes that correspond to all the computation stages of the application. Each $\tau i \in T$, $1 \leq i \leq n$ represents a computation stage of the DAG. E is a set of communication stages, interconnecting the computation stages. L represents the deadline of the DAG. A computation stage consists of tasks that can be executed in parallel such as a map and reduce.

The inter-coflow scheduling problem has been proven to be NP-hard. Thus, Genetic Algorithm (GA)-based Coflow scheduling to guarantee the deadline of DAG.
The user intends to run a BoT scientific application consisting of a set of n CPU-intensive independent tasks, that n is known a priori, and that all tasks are ready for processing at time t =0. The system provides a list of all possible Pareto-optimal points and lets the user choose a desirable resource allocation based on his/her desired utility function. To align our idea with reality, we argue that Pareto-optimal points with an integer completion time are a good candidate set to deliver to the user(Y. Yang et al(2014).
DPDS is an online algorithm that provisions resources and schedules tasks at runtime. It consists of two main parts: a provisioning procedure, and a scheduling procedure. Workflow-Aware DPDS - This is an extension of the DPDS in which introducing a workflow admission procedure, which is invoked whenever WA-DPDS sees the first task of a new workflow at the head of the priority queue(M. Malawski et al(2012)). Further directions can also be approached using Autoscalingfeature(M. Mao et al 2011).

III. METHODOLOGY

A. Deadline-constrained Probabilistic List

The proposed approach is used to schedule the jobs for minimizing the cost in two steps. The proposed algorithm builds an ordered job list and then allocates each job from the list of services. The proposed approach focuses on deadline constrained cost optimization and so it needs one more step that is distributing the whole deadline to each job of the workflow. Then in the service selection step, the proposed algorithm allocates each job to the particular service based on the sub-deadline criteria in order to minimize the cost.

B. Deadline Distribution Based on Probabilistic Upward Rank

Consider the upward rank of a task this the longest path. The proposed algorithm chooses the fastest service because the IaaS services are infinite and are heterogeneous in clouds. The proposed algorithm first takes the scheduling problem to satisfy the deadline. Through traversing the graph upward, the upward rank can be computed. The length of the workflow's critical path is denoted as an upward rank of the entry.

Figure 1 shows the architecture of the proposed work. According to the proposed work, users are requested to submit their tasks and the needed resources to the request manager through a user interface. The task request must contain the size of the task, amount of data needed, deadline value and the cost. After analyzing these details the task manager sends the request information to the scheduling manager. The scheduling manager is taking the responsibility of allocating the tasks. The scheduler deploys all scheduling methods such as all types of task performance priority strategies under deadline and cost constraints. The public and private cloud information are stored in the scheduling manager. The scheduling manager has information like computing power and energy consumption of the public and private resources. Also data transport capacity and computing cost details are maintained by the scheduling manager. Now the scheduling manager allocates a task to public or private cloud resources based on the demands and combines resource information. The computational results are then given to the users.

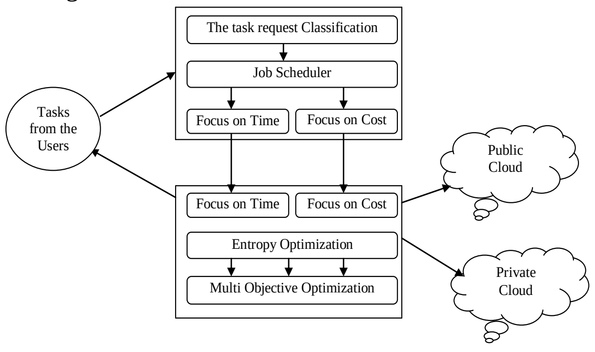

Fig.1. Multi-Objective Optimization-based Cloud Model

A. Task Ordering and Service Selection

Even though there are lots of methods available to allocate the jobs to the virtual machines in the cloud, in this work the proposed probabilistic upward rank is taken to order the jobs in the available virtual machines. The service methods have all the services and jobs which have to be allocated with the virtual machines. The service selection is referred as to select a service that satisfies its sub-deadline and minimizes the cost of data transmission. Consider if a service doesn't meet the sub-deadline then the job is not allocated to the virtual machines. Also if the service is not of the fastest type then the service is made to be a faster level and update the finish time of each job deployed on it. In addition to that the leasing period of each task starts when the first task begins to receive the data from the virtual machines.

B. Working Mechanism

The scheduling process is the process that defines a virtual machine as a workflow and its respective tasks. The dimension of the virtual machine will be equivalent to the count of the tasks in their corresponding workflow. This dimension will provide the coordinate system that can be used to locate the position in space. DCRA is initialized with a group of random solutions and then searches for optima by updating generations. In every iteration, each virtual machine is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. This value is called best. DCRA does not have genetic operators like crossover and mutation. VMs update themselves with the internal velocity. They also have memory, which is important to the algorithm. The range of the movement of a virtual machine is calculated by the resources available for running the tasks. The logic will be applicable to the remaining coordinates and their corresponding values. By considering the above scenario, the elasticity and dynamicity of the resource acquisition model, no initial set of available resources can make use of the input in the algorithm.
The artifice is to determine the initial stage of the resources that an algorithm uses to explore the solutions and attain the scheduling intention to reflect the position of the particle in its own place. Such artifice needs to be reflected in the diversely populated particles and provide necessary DCRA options to obtain the suitable solutions that are produced. Hence, the problem is a population-based stochastic approach for solving continuous and discrete optimization problems. To compute the virtual machine's position into a schedule, make the group of resources lease 'L' and their corresponding tasks using Map reduce as 'R' as empty initially.

The total execution cost and the time is set to zero. The algorithm executes the time of the workflow for every task performed by its respective resources. They are represented in the matrix which defines the row being' as the tasks and the columns 'n' define the resources. 'Tn' represents the time taken to run a task 'Tm' on the resource 'Lm'.

The algorithm gets the required values and data items that are necessary to start the decoding of the location of the virtual machine and build the schedule. To attain this, each virtual machine will iterate through every co-ordinate m in its location and updates the L and R. It obtains that which task and resource are being used in the current coordinate and the values. This is done by encoding the artifice defined earlier. Therefore, it will start to run as soon as the resource is assigned to its available value. Incase if the resource is busy with some other task at the same time, then the execution will be delayed until the VM is set to free for the further execution T(m). The value is calculated over the total processing time and the start time of the task.

The first objective is the respective time for the VM to determine when the execution has to be launched and if the resource is already present then it states that the resource has its start time and does not require to be updated. If it is new for a resource then the first task will be to assign them a start time of the task to boot the VM's. The second objective is the updated time of when the resource has been scheduled to finish the execution that has been launched for a particular task to either shut down or assign another task to run after the completion of one task at a time. Along with this the whole task is to map a resource tuples and assign them as estimated start and end time.

A workflow of the resource is modeled by a Directed Acyclic Graph. Each node will represent a particular task and the edges will denote the transfer time of data between the tasks. A child's task will not execute until the parent's task has been completed its execution and the dependencies, data, and control. A deadline is explained as the time limit specified by the user for a particular task has been met or not. All the computational and storage services are assumed to be in the same data region hence the bandwidth for the average services is roughly set to be equivalent. Computational services provide the different varieties of virtual machines with their CPU Type configurations available at different price ranges. There is no limitation to how many numbers of VM instances used by the services. The time interval will be provided by the cloud service provider for the execution of the workflow and the data transfer cost is estimated to be zero and free for internal transfers, though the real cloud providers will charge us for the storage of data items externally based on the volume of the data.

Virtual machines use three major modules such as resource provisioning module, workflow scheduling module and the executive manager for smoother performance of the task with all the constraints achieved. This is to analyze the structure of the workflow for estimating the number of resources needed for the execution. Once the workflow is scheduled, the execution manager will identify the resource and map to the provisioned tasks in the workflow. The comparison between the computing model and the other high-performance models states that the resource is allocated in a workflow-driven by the user predefined and the resource size is set based on the variation during the execution of the process.

C. Virtual Machine Allocation Procedure

In cloud computing, resource allocation is the biggest issue involved in allocating the resources to corresponding tasks. The proposed method in this work is an allocation technique that provides high quality allocation for resources by considering threshold levels of the datacenters. The agent provides authentication to each user to ensure the privacy of the system.

The algorithm is determined to minimize the overall energy consumed and determining the satisfied strict timeline. A task scheduling algorithm will initialize the scheduling scheme to optimize the total execution time and its cost. A task can be migrated to a cloud service while the deadline of the application is having reduced energy-consuming. The execution of a task must be of optimal power-consuming to save energy of remote execution with the help of an integer linear programming method.

It defines the motive to attain the minimal execution cost on a user-specified deadline which is given as a constraint. The execution order of every task is obtained by prioritizing them and choosing the selected task to be executed based on the minimal trade-off of the total cost and the finish time. The dominance of the relationships and the distance of the crowds are required to guarantee the optimization of the makespan and the execution cost.

The total number of workflows concentrated by the resource on its task is optimized which is the major objective to schedule a task. The DCRA algorithm is to address the scheduling issue with the help of a cloud model to maximize the total profit that meets the capacity of the resource of every service provider. The major focus on elaborating the DCRA algorithm is to solve the multiple objectives and also addressing the constraints efficiently.The overall resource allocation is achieved by proposed method by performing K-means clustering based average median absolute derivation. The average median absolute derivation ~~is~~ efficiently calculated the threshold levels of the datacenters. Based on these threshold levels, the datacenters are classified into four classes and then the virtual machines are allocated to their corresponding datacenter based on their threshold levels. The migration process is performed by considering the minimum level of memory consumption and CPU consumption. The resource allocation process is mainly affected by network traffic and storage capacity. In cloud system, similar dependencies appear between the resources that lead to network traffic when number of clients processes the same resources. It causes higher energy consumption in the resource allocation process.

Datacenters in the cloud environment allocate the user's tasks to the virtual machines. Network traffic and storage facilities are main issues in the allocation of VM in cloud environment. Due to high network traffic, more amount of energy is consumed for resource allocation, which is the main cause of poor performance of the system. To solve this problem, a new solution is prepared that allows reducing energy consumption while allocating the resources. During the resource allocation tasks are allocated to the available resource without loss of energy.

In this work, the method to improve energy efficiency in the resource allocation process has been proposed. Generally, cloud infrastructure receives huge amount of tasks from a large number of user. All tasks are received by the broker in the cloud system. Broker performs resource allocation based on the proposed technique. In this technique, initially the threshold levels of the datacenters are calculated by using K-means clustering based average median absolute derivation. In this process, three threshold levels of datacenters are calculated such as low threshold, moderate threshold, and high threshold. Based on these calculated threshold values, the datacenters are classified into four types such as little loaded, lightly loaded, moderate loaded and high load. The virtual machines are allocated to datacenters by considering their threshold levels. The VM migration process is carried out by considering minimum memory consumption and minimum CPU consumption of the datacenters.

To provide efficient resource allocation, the cloud environment is comprised of users, agents, brokers, and datacenter. The multiple numbers of user tasks are received by the agent. Then the received tasks are transformed to brokers to achieve optimal resource allocation. The broker applies the proposed method to provide load-balanced placement to virtual machine and also responsible for migrating the tasks from overloaded virtual machine to idle virtual machine. In the proposed method, the agent is responsible to provide authentication to each user who is involving to allocate their tasks to resources. The proposed system allows only legitimate users who are authenticated by agent. Then the broker receives the tasks from the authenticated users to provide load-balanced placement. In the proposed system, the migration process is achieved by considering the minimum memory consumption and minimum CPU consumption. It improves the efficiency of the system by migrating the tasks. If the virtual machine is overloaded by tasks, the load-balanced migration process is performed to migrate the tasks to idle virtual machines. Then the energy consumption is minimized by shut down the idle physical machines in the host.The main goal of the performance evaluation is to analyze both the quality and performance of the proposed method under different load conditions. The performance of the proposed system is evaluated by using various metrics such as execution time and threshold levels which provides better results compared to the existing system.

According to Amazon's cloud service, the public cloud resource (R_u) and the private cloud resource (R_r) are defined in Equation (1) and Equation (2) respectively.

$R_u = < C_u, L_u, P_u, S_u, Q_u>$ (1)

$R_r = < C_r, L_r>$ (2)

where C_u and C_r are the computing power of public and private cloud resource respectively. L_u and L_r are the transmission capacity public and private cloud resource respectively. P_u is the computing cost, S_u is the data storage cost and Q_u is the transmission cost. The private cloud considers only the computing power and transmission capacity.

Now a task T_i can be assumed as a task request from the users and each resource takes only one task at a time. Then T_i can be defined as given in Equation (3),

$T_i = <TD_i, TC_i, Ta_i, TM_i>$ (3)

where TD_i denotes the deadline of T_i where T_i has to be finished and the results have to be returned prior to TD_i. If this does not happen then T_i is considered a failure. TC_i denotes the size of T_i. TL_i denotes the amount of data associated with T_i and TM_i denotes the budget cost of T_i.

There are two deadline constraint conditions are taken in task scheduling which is referred to as $TD_i$and TM_i. To meet the above two deadline constraint conditions, it is required to compute the task completion time and TM_iunder the scheduling model of the hybrid cloud. There are two parts in the task-completion time which is termed as computing time and transmission time. The task completion time in the public cloud is termed as t_iR_u and private cloud is termed as t_iR_r. There are two deadline constraints which are shown in Equation (4) and Equation (5) respectively.

$t_iR_r \leq TD_i$and $t_iR_u \leq TD_i$where

$t_iR_u = \frac{TC_i}{C_u} + \frac{TL_i}{L_u}$ (4)

$t_iR_r = \frac{TC_i}{C_r} + \frac{TL_i}{L_r}$ (5)

The cost related to scheduling in the hybrid cloud is mainly the cost of public cloud services which consists of P_r, S_r, and Q_u. The cost which is needed to complete Ti in a public cloud is computed as shown in Equation (6).

$F_i = TC_i * P_u + TL_i * Q_u + TL_i * S_u$ (6)

IV. PERFORMANCE EVALUATION OF DCRA

The proposed methodology is a virtual machine placement which aims at enhancing the reliability of server-based cloud services whose fault-tolerance level can be measured and assured in terms of the k-fault-tolerance metric. 32-port fat-tree data center network is constructed in the experiment as shown in Figure 2. 10Gbps is set as the capacity of the root-layer link and aggregation layer link and 1 Gbps is set as the capacity of the edge layer link. There are 16 host servers in each subnet. Each of these host servers can host four VMs at most. The performance analysis of the proposed method is implemented and studied.

The algorithm to estimate the meeting of predefined user deadlines is plotted for each of the workflow tasks and the deadline intervals. Constraints can be separated into four categories: time, execution cost, availability of the resource and the network factors. Data availability is also related to the evaluation timeline. The performance of the PSO, Particle Swarm Optimization Homogeneous (PSO_HOM) and Scaling-Consolidation-Scheduling (SCS) is similar to each other from the deadline intervals. When compared to these, DCRA has performed better upon these algorithms.

The deadlines are met by differing the counterpart and capturing the dynamicity of the performance variation. This needs to be adapted to the conditions of the cloud to make sure of the deadlines met. PSO_HOM and SCS have met most of the deadline for all its tasks performed by every resource workflows to make them look appealing to the algorithms for the optimal scheduling.

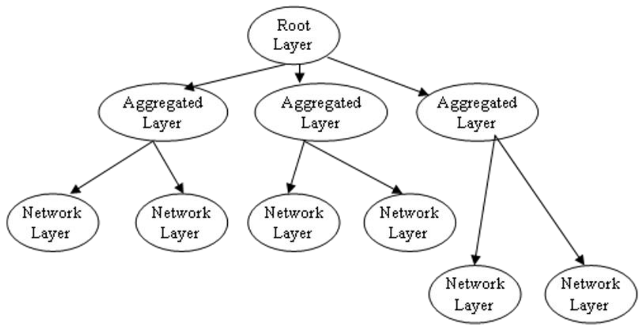

Fig.2. Description of Fat Tree Structure

In the DCRA algorithm, the dynamic characteristics of the cloud and the variability of the overall CPU performance are considered when compared to other three algorithms as the difference in the percentage of the evaluated deadline is met by DCRA.

The proposed approach is evaluated in terms of total transfer time, time complexity, aggregated performance, root layer network consumption, network layer consumption and the normalized average cost per job.

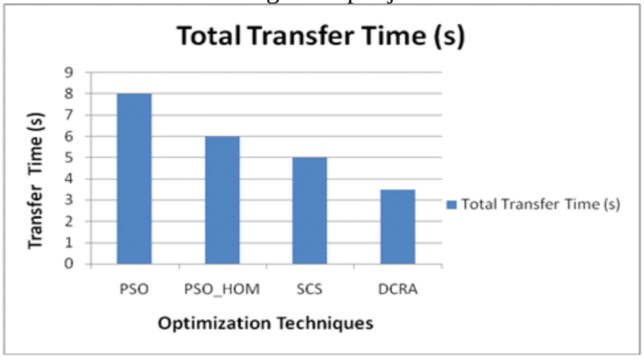

Fig.3: Performance Analysis of total transfer time

From Figure 3, it is observed that the total transfer time of data between the virtual machines to the host server takes less time when compared to the existing approaches. The existing approaches like PSO, PSO_HOM, and SCS are taken for the comparisons with the proposed approach.

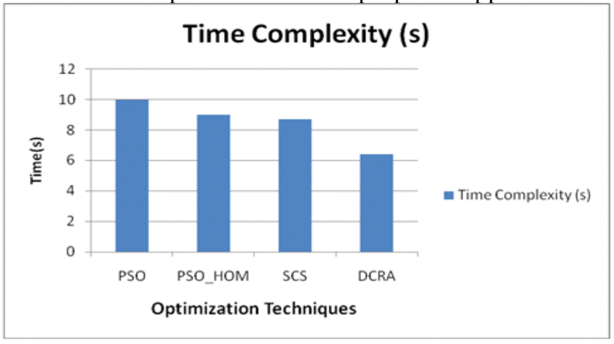

Fig.4: Performance Analysis on Time Complexity

Time complexity is referred to as the total time needed to get the final result. From Figure 4, it is observed that the time complexity of the proposed approach is 27% low when compared to the existing approaches. It is because of the multiuser execution system of the proposed approach.

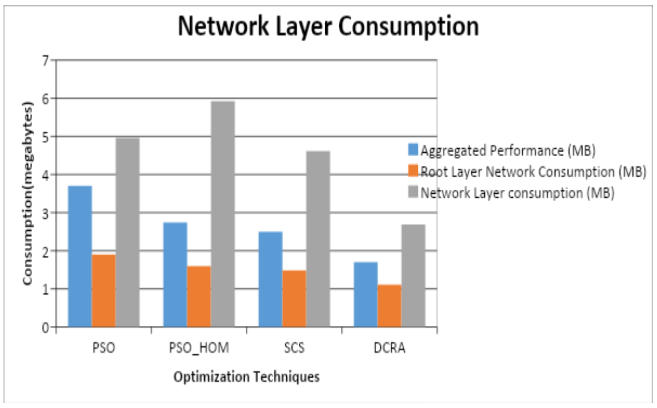

Fig.5: Performance Analysis on Root, Aggregated and Network Layer Consumption

From Figure 5, it is observed that the proposed approach gives a decrease in aggregated performance in terms of megabytes. The proposed approach gives low aggregated performance when compared to the other existing approaches by 40%. It is observed that the proposed approach gives a decrease in root layer network consumption (35%) in terms of megabytes. Also, it is found that the network layer consumption is 45% low when compared to the existing approaches.

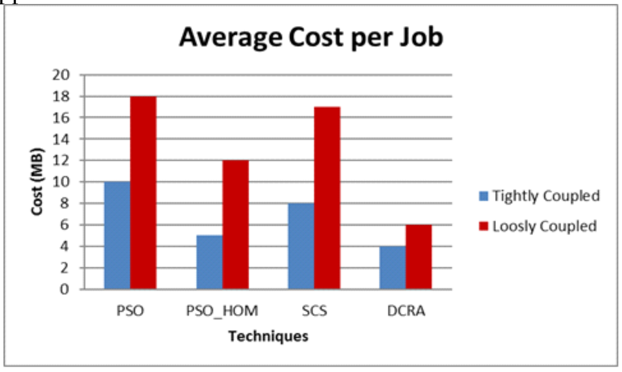

Fig.6: Performance Analysis on Normalized Average Cost per Job

From Figure 6, it is observed that the proposed approach gives a decrease in normalized average cost per job. The normalized average cost per job is evaluated in terms of tightly coupled and loosely coupled machines. The average cost per job in tightly coupled devices is 25% low for the proposed approach when compared to the existing approaches. The average cost per job in loosely coupled devices is 35% low for the proposed approach when compared to the existing approaches.

V. CONCLUSION

The proposed approach gives the deadline for each job and performs a two-step optimization scheduling algorithm. The proposed algorithm minimizes the cost where the jobs are ranked and each job is allocated to a service in which the sub-deadline is met. Based on the pheromone trail and probabilistic upward rank, the ant builds an ordered job list and uses the same deadline and service selection methods to construct the solutions. From the experimental results and performance evaluation, it is observed that the proposed algorithm works well when compared to the existing approaches.

REFERENCES

1. Abaker, I, Hashem, T, Yaqoob, I, Badrul, N, Mokhtar, S, Gani, A &Ullah, S 2015, ‘The rise of big data on cloud computing: review and open research issues’, Information Systems, vol. 47, pp. 98-115.
2. Kumar Mohit& Sharma, SC 2018, ‘Deadline constrained based dynamic load balancing algorithm with elasticity in cloud environment’, Computers & Electrical Engineering, vol. 69, pp. 395-411.
3. Liu, L, Qi Fan &RajkumarBuyya 2018, ‘A Deadline-Constrained Multi-Objective Task Scheduling Algorithm in Mobile Cloud Environments’, IEEE Access, vol. 6, pp. 52982-52996.
4. Chen, C, Lin, J &Kuo, S 2018, ‘MapReduce Scheduling for Deadline-Constrained Jobs in Heterogeneous Cloud Computing Systems’, in IEEE Transactions on Cloud Computing, vol. 6, no. 1, pp. 127-140.
5. Malawski, M, Juve, G, Deelman, E &Nabrzyski, J 2012, ‘Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds’, SC ‘12: Proceedings of the International Conference on High-Performance Computing, Networking, Storage and Analysis, Salt Lake City, UT, pp. 1-11.
6. Mao, M & Humphrey, M 2011, ‘Auto-scaling to minimize cost and meet application deadlines in cloud workflows’, in Proceedings of the International Conference for High-Performance Computing, Networking, Storage and Analysis (SC), ACM, p. 49.
7. Patidar, Shyam, DheerajRane, &Pritesh Jain 2012, ‘A survey paper on cloud computing’ Second International Conference on Advanced Computing & Communication Technologies,IEEE, pp. 394-398.
8. Moens H, Handekyn, K& De Turck,F 2013, ‘Cost-aware scheduling of deadline-constrained task workflows in public cloud environments’, IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), Ghent, pp. 68-75.
9. Ruiz-Alvarez, I, K, Kim& Humphrey, M 2015,’Toward Optimal Resource Provisioning for Cloud MapReduce and Hybrid Cloud Applications’, IEEE 8th International Conference on Cloud Computing, New York, pp. 669-677.
10. Yang, Y, Xu, J, Wang, F, Ma, Z, J. Wang & Li,L 2014,’A MapReduce Task Scheduling Algorithm for Deadline-Constraint in Homogeneous Environment’, Second International Conference on Advanced Cloud and Big Data, Huangshan, pp. 208-212.
11. Xu, M, Alamro,S, Lan, T &Subramaniam, S 2017,’LASER: A Deep Learning Approach for Speculative Execution and Replication of Deadline-Critical Jobs in Cloud’, 26th International Conference on Computer Communication and Networks (ICCCN), Vancouver, BC, pp.1-8.