Deadline and Cost based Workflow Scheduling in Hybrid Cloud

Nitish Chopra
UIET, Panjab University
Chandigarh, India
nitish_chopra2002@yahoo.com

Sarbjeet Singh
UIET, Panjab University
Chandigarh, India
sarbjeet@pu.ac.in

Abstract— **Cloud computing provides on demand resources for compute and storage requirements. Private cloud is a good option for cost saving for executing workflow applications but when the resources in private cloud are not enough to meet storage and compute requirements of an application then public clouds are the option left. While public clouds charge users on pay-per-use basis, private clouds are owned by users and can be utilized with no charge. When a public cloud and a private cloud is merged, we get a hybrid cloud. In hybrid cloud, task scheduling is a complex process as jobs can be allocated resources either from private cloud or from public cloud. Deadline based scheduling is the main focus in many of the workflow applications. Proposed algorithm does cost optimization by deciding which resources should be taken on lease from public cloud to complete the workflow execution within deadline. In the proposed work, we have developed a level based scheduling algorithm which executes tasks level wise and it uses the concept of sub-deadline which is helpful in finding best resources on public cloud for cost saving and also completes workflow execution within deadlines. Performance analysis and comparison of the proposed algorithm with min-min approach is also presented**.

Keywords—**Cloud Computing, Workflow Scheduling, DAG, Private cloud, Public cloud, Hybrid cloud**

I. INTRODUCTION

Cloud computing is a model which provides on demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. Cloud computing is being used to deliver on demand storage and processing power. This environment allows the leasing of resources to improve the locally available computational capacity, supplying new computing resources when necessary. In a cloud, the user accesses computing resources as general utilities that can be leased and released.

Virtualization is the process of presenting a logical grouping or subset of computing resources so that they can be accessed in abstract ways with benefits over the original configuration [2]. It allows abstraction and isolation of lower level functionalities and underlying hardware [3]. The virtualization software abstracts the hardware by creating an interface to virtual machines (VMs), which represents virtualized resources such as CPUs, physical memory, network connections, and peripherals. [4] Each virtual machine alone is an isolated execution environment independent from the others. Each VM can have its own operating system, applications and network services. Virtualization allows server consolidation, hardware normalization, and application isolation in the same physical machine. Virtualization technologies, such as Xen [5], are an important part of a cloud computing environment, since the cloud provider can dynamically offer different hardware and software configurations to the cloud user. With this, the user can select a machine configuration and manage this machine with full privileges without interfering in cloud machines available to the others.

We can classify cloud in three different types on the basis of availability of resources

Private clouds: Private cloud is also known as internal cloud, these are designed for exclusive use by organization as the organization owns the resources. Once a private cloud is deployed, the resources can be accessed as according to the need of an organization but only limited number of resources are available in private cloud.

Public clouds: In public cloud resources are provided on per usage basis, so the user has to pay for the time he/she is utilizing those resources.

Hybrid clouds: It is the composition of private and public cloud. If the resources provided in private cloud are not sufficient then the resources are taken on lease from public cloud to complete the given task. Considering the resource availability issue, hybrid cloud is considered as best for an organization because it has advantages over both private cloud and public cloud. Private cloud is owned by an organization but it has limited access to resources as most of the times the resources taken in private cloud are not enough to complete the execution of all tasks whereas only public cloud resources are sufficient but monetary costs must be kept in consideration as it may be out of budget of an organization to take all resources prior on lease from public cloud. So to use hybrid cloud is the best option because it is the composition of private and public cloud, it will take resources on lease from public cloud only if private cloud resources are not sufficient. Hybrid clouds provide the elasticity feature to the computing environment of user.

Task scheduling involves process of assigning tasks to available resources on the basis of task characteristics and

978-1-4673-6217-7/13/$31.00 ©2013 IEEE

requirements. Task scheduling is a necessary process as it schedules tasks on the basis of the requirements of the user. One requirement can be to minimize completion time to complete execution within deadline by reducing the monetary cost. This requirement of user is considered throughout this paper.

Workflows are represented by DAG (directed acyclic graph) in which nodes represent tasks and the edges represent the dependencies. The dependencies show that output of one task may act as an input to other task. Seven node DAG is shown in Fig.1. in which tasks on level 2 cannot start their execution until level 1 tasks has completed its execution . Similarly level 3 tasks cannot be executed until level 2 tasks have completed their execution. This is because of the dependency between level 2 and level 3 tasks. So task scheduling is a complex process because of the dependencies involved in it. All the simulations have been done on DAG shown in Fig.1.In this paper, the problem of scheduling tasks is addressed to complete the workflow execution within deadline with minimum cost.

II. RELATED WORK

There are few techniques given to schedule independent tasks in hybrid environment like in modified bees life algorithm [6] in which independent tasks are scheduled in hybrid environment to minimize the makespan . The main objective of scheduling technique is to minimize makespan of the workflow but there are number of task scheduling algorithms which consider cost and deadline as factor for scheduling as well. Van den Bossche [7] presented scheduling technique which schedules tasks for cost optimization and achieves output within deadline. There are several techniques which consider both cost and deadline as main parameters for scheduling dependent tasks. Buyya and Yu [8] proposed work which uses genetic algorithms to schedule workflow applications on computational grids. In another work Buyya presented a technique [9] to schedule workflow applications using sub-deadlines and MDP algorithm but it does not consider dynamic leasing of resources. Luiz has presented HCOC [2] algorithm which allows dynamic leasing of resources as well , it uses HEFT [10] or PCH [11] to make initial schedule in private cloud and uses HCOC algorithm for scheduling in public cloud. It minimizes cost and completes workflow execution within deadline as well. A survey of workflow scheduling algorithms is also presented in [12]

III. WORKFLOW SCHEDULING FOR COST OPTIMIZATION WITHIN DEADLINE IN HYBRID CLOUD

User defined deadline of a workflow is the time by which the workflow execution must be completed. The output must be achieved before the given deadline else the output would be of no use. In workflows, makespan is described as the total time taken to complete the execution of all tasks of workflow. In a workflow represented by DAG, the time at which the output of last node is available to user is known as makespan of a workflow.

A private cloud is user owned cloud so in private clouds cost is not a concern for the user. The only concern when executing tasks on private clouds is that execution of all tasks should be completed within its set deadline. But in case of big workflow applications private cloud with limited processing power may not be sufficient to execute all the tasks within the set deadline, so the need of acquiring resources from public cloud arises. Proposed work focuses on scheduling in hybrid cloud i.e. if execution is not done within deadline in private cloud then resources would be taken from public cloud to complete execution within deadline. We may run the part of the application on public cloud to achieve output within deadline as public cloud resources has much high processing power as compare to private cloud resources. Along with deadline a new factor i.e. cost needs to be considered while taking resources on lease from public cloud, but running whole application on the public cloud can be very costly. So, the proposed algorithm considers deadline and monetary cost as the main factor for scheduling tasks and resources in hybrid cloud environment. DAG for proposed work is given below in fig.1.

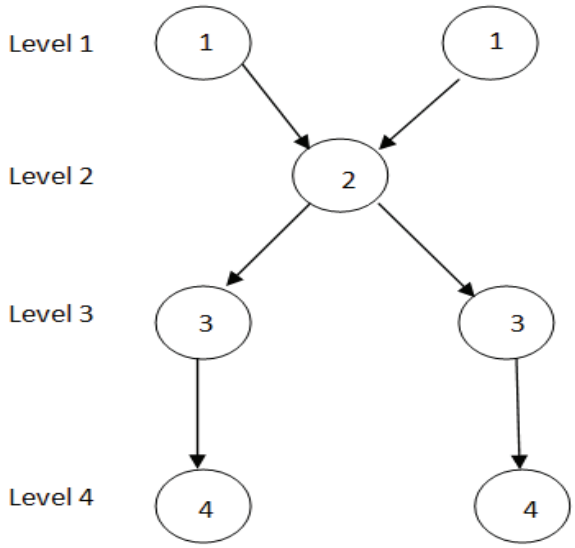

Fig.1. An example of DAG with level based scheduling

As a workflow application is divided in number of dependent tasks and larger the number of instructions in the task, more will be its execution cost. So our algorithm tries to schedule the heavy tasks on the private cloud and if a task may not be executed on private cloud then task is scheduled on the public cloud. The main idea is that if heavier tasks will be executed on private cloud resources then it will be cheaper to execute lighter task on public cloud. Therefore along with managing the cost, proposed algorithm also makes sure that application should be executed within its set deadline.

Deadline of complete workflow is given by user and sub-deadline i.e. the deadline for each task is calculated based on the deadline of workflow defined by user. We find the sub-deadline for the task from the application deadline using a percentage method.

Sub-deadline for the task = percentage of share of task in application * deadline of the application + deadline of task's predecessor
For root node i.e. the node which does not have a parent, the deadline of task's predecessor would be set to 0 in that case.
This sub-deadline i.e. individual deadline for each task is calculated prior to scheduling algorithm.

Following steps explain the proposed algorithm in brief:

Step wise algorithm.

1. Proposed algorithm is a level based scheduling algorithm which means that the whole workflow application is divided into number of levels where tasks at each level will be executed in parallel and are not dependent on each other. Scheduling of the tasks is done at each level. Level based scheduling allows the better allocation of VMs. As at the end of each level, all VMs are finished with the execution of the tasks of the current level. So for the next level all the VMs are available for the allocations to the tasks and we may schedule best resources to the new level tasks.

2. Each VM in proposed algorithm maintains a list of running tasks. That list contains the number of tasks that VM is running or will run in the current level. While running multiple tasks on the same VM, execution time of each task may increase so proposed algorithm find the total estimated execution time of the tasks on the VM and make sure that it must be less than the deadline of each task so that each task must be finished within its sub-deadline.

3. In this proposed algorithm, there are two phases of scheduling. First phase is scheduling at private cloud and second is scheduling at public cloud.

IV. Algorithm for Scheduling In private Cloud

In private cloud scheduling algorithm we first try to schedule the heaviest tasks so the most costly task should run on private clouds. This approach is used because private cloud resources are owned by user and considered as free to use.

Following is the pseudo code for scheduling in private cloud

1. For initialization firstly submit the list of tasks T={t1,t2,t3,…….tn}, to be scheduled, and list of private VM's VM(Private)={v1,v2,v3……..vm} to the scheduling algorithm where i=1….n and j=1….m.

2. Sort the list of tasks i.e. T on the basis of number of instructions (ni) and list of VMs i.e. V (Private) on the basis of processing power (Million instructions per second), in descending order.

3. As proposed algorithm is level based algorithm. Each task has a field pointing to its level. And number of levels in the *application can be represented as L={l1,l2,l3,……,lk};*

4. Pick all tasks from level(k) and put them is a temporary list templist.

5. Now templist tasks will be scheduled. First heaviest task i.e. t(i) in the level will be picked from templist and algorithm finds the best VM for it in next step.

6. Find the best VM for the chosen task, VM with lowest execution time will be the best VM for the task. Algorithm also considers the fact that VM may execute more than one task. Therefore in proposed algorithm each VM maintains a list known as RTL (running task list) which contain the tasks running in current level.

As multiple tasks will share VM, execution time of each task may increase. Therefore the proposed algorithm checks that total execution time of the tasks on the VM should be less than the deadline of each task.

LET (i)=ni(i)/mips(j)

Where LET(i) is lowest execution time for task i which we are trying to schedule and it is heaviest task in the templist.
ni(i) is number of instructions in task i.
mips(j) is processing speed of VM(j).

if(RTL(i)!=null)
then
TET= PST(1,j)+PST(2,j)+ .....PST (k,j) +LET (i,j)
Else
TET=LET (i,j)
{All PST values can be 0 in two scenarios
i) If we are trying to schedule the first task on a particular VM or
ii) after comparing TET with sub- deadline in below step i.e. if any other task cannot be scheduled on this particular VM then only current task is tried to be scheduled on that particular VM}
PST (k,j) is the execution time of the already scheduled tasks t(k) assigned to particular VM(j) where k=1,2…..n
j is the chosen VM
i is the current task which is to be scheduled
TET is total execution time of all the tasks already scheduled on chosen VM (j) in current level.

7. Deadline for the individual task known as sub-deadline of the whole workflow deadline i.e. d(i) is compared with LET(i) and other tasks assigned to VM(i).

Loop until end of RTL(i):
Compare sub-deadline of each task in the RTL(i) with TET(i) and

If

TET(j)<d(1,2,……..,i) where for i=1,2…n tasks t(i) will assigned to the VM(j).

Once task is assigned to the VM (i), update the list of task running on VM.

Else

TET(,j)>d(i) then the task t(i) will be added to the list of the tasks that will be tried to schedule to next VM , this continues till all the VMs in private cloud are allocated

Where TET is total execution time for the all the tasks running on VM(j).

d(1,2,……..,i) is deadline for all the tasks in TList(i).

8. Update level in step 4 and repeat steps 5 to 7 till last level.

In the first step, tasks list and VM list are both submitted to the scheduling algorithm. Here the tasks list contains all the tasks of the workflow and VM are the virtual machines which are available in private cloud . In the 2nd step, tasks are sorted on the basis of computation cost which is based on the number of instructions required to complete the task execution. VM's are sorted on the basis of processing power of VM in descending order. 3rd step describes that algorithm works level wise , for every task there is a field which is pointing to its level where L is the list of levels l1,l2 etc. In the 4th step, tasks which belong to a particular level are put in temp list for example for first iteration i.e. for i=1, all the tasks from level 1 are put in temp list. In the next step, tasks from the templist are scheduled with the heaviest task being chosen first, here the heaviest task is the one which requires most computation (number of instructions) among others in the list. In the 6th step, best VM is chosen for the heaviest task from templist i.e. the VM which is having the highest processing power is chosen. As private cloud resources are considered free to use , the VM with highest processing power is chosen first. In this step it is also considered that a VM can execute more than one task, a list known as RTL (running task list) is maintained by every VM which contains the tasks running in current level. In this step it is also checked that total execution time of tasks on the VM should be less than the individual deadline (sub-deadline) of a task. LET(i) is the lowest execution time for the heaviest task which we are trying to schedule on that particular VM , it is calculated on the basis of number of instructions and MIPS of VM . It is called the called the lowest execution time because the VM with highest processing power is chosen first from private cloud resources. After this, TET(i) i.e. total execution time is being calculated by adding LET(i) to the execution time of previously scheduled tasks on that particular VM. Please note that for first iteration every PST value will be 0 as there is no already scheduled task on that VM. In the 7th step, total execution time is being compared with each task individual deadline (sub-deadline) and if each task completes its execution within their calculated sub-deadline then the tasks are allocated to that VM but if the sub-deadline of a task is violated then it will be tried to allocate to the next VM, this continues till every task get a VM which completes execution within sub-deadline. But if sub-deadline is violated for all VM's then it will be added to the list of tasks which are to be scheduled on public cloud resources. In the last step 8, level is updated which means the same process (7 steps) is repeated at each level.

V. Algorithm for scheduling in Public Cloud

First all the tasks are scheduled on private VMs. Tasks that cannot be executed on the private are added to a list called public task list. This list is given to the public scheduling algorithm. Public scheduling algorithm finds the best VMs for the tasks. According to the proposed algorithm, best VM will be the one that is cheap and will do our job in the given deadline. In public scheduling algorithm, cost factor is added. In our scenario we are considering monetary execution cost. Execution cost is set on the basis of machine processing power i.e. high processing VM will cost more than the low processing machine

Following is the pseudo code for scheduling in public cloud:

1. Submit the list of tasks T={t1,t2,t3,……tn}, to be scheduled, and list of public VMs VM(Public)={v1,v2,v3……vm} to the scheduling algorithm where i=1,2,3….n and j=1,2,3…m

2. Sort the list of tasks i.e. T on the basis of number of instructions (ni) in descending order and list of VMs i.e. V (Public) on the basis of cost in ascending order. Sorting VM (Public) on the basis of cost will help in picking lowest cost VM first.

3. Make a list of all tasks in current level. TempList(k) will contain all the tasks in level k.

4. Task t(i) is picked from the list and its estimated execution time EET(i) and execution cost EEC(i) is calculated on the VM(j).

Repeat steps 5 and 6 for j=1 to m {All VM's are considered for allocation}

5. Public VMs maintain a list of running tasks, RTL(Running Task List).

if(RTL(i)!=null)

then

TET(j)=PST(1,j)+PST(2,j)+ ……..+EET(i).

Else

TET(j)=EET(i).

6. Compare sub-deadline of the each task in the running task list with the total execution time. If total execution time of tasks on chosen VM(j) is less than sub-deadline then check if estimated execution cost of the task on VM(j) is less then cost of executing it on next VM in list known as VM(j+1). If VM (j) cost is less than the next VM (j+1) then the task is assigned to VM (j). Similarly this process is repeated till all the available VM's are evaluated for assigning.

If (TET(j)< d(1,2,……..,n)) where 1,2,……..,n are tasks in RTL.

And

If (EEC(j)<EEC(j+1) assign task to VM(j)

Else if (EEC(j)>EEC(j+1) assign task to VM (j+1);

Else update value of j, calculate EEC (j), EEC(j+1) and repeat steps 3 to 6

Where TET(j) total execution time of all the tasks assigned to VM(j).

7. Update level and repeat steps 3 to 6 till last level.

For scheduling in public cloud, in first step the task list contains the tasks which needs to be scheduled on public cloud resources and VM list contains the VM's available from public cloud. In 2nd step tasks are sorted in descending order according to the computation (number of instructions) required to execute the task whereas VM's are sorted in ascending order according to the cost of public cloud VM's. In the 3rd step, tasks from a particular level are added to templist . Now in 4th step one task is picked in each iteration and its estimated execution time (EET) and estimated execution cost(EEC) is calculated. In 5th step TET (total execution time) is calculated by adding the PST(1,j)…(k,j) and EET(i) where PST(1,j)…PST(k,j) is the execution time of previously scheduled k tasks on particular VM(j) and EET(i) is the current task i which we are trying to schedule on this particular VM(j) . In the else statement TET(j)=EET(i), it describes that estimated execution time of current task t(i) is taken as total execution time on VM(j) , this is assigned in first iteration when we are trying to schedule the first task as there are no tasks which already scheduled on that VM . In the 6th step, TET(total execution time of all tasks) on a chosen VM (j) is compared with the sub-deadline of each task, if TET is less than sub-deadline then cost of the chosen VM(j) is compared with cost of next VM(j+1) and VM with minimum cost is chosen . In the else portion the value of j is updated and similarly cost of next chosen VM is compared with its next VM .This process continues till the VM with minimum cost which completes the task execution within defined sub-deadline is found and allocated to tasks of that particular level. In the 7th step, level is updated that is steps 3 to 6 will be repeated for each level.

VI. RESULTS

Proposed algorithm is very exciting and challenging to implement. For experiments we have used cloudsim simulator [13]. In the simulations we evaluated the algorithms using different cloud sizes, partial workflow application, cost and deadlines. Results are based on makespan and cost of execution of workflow application on public cloud, hybrid cloud with level and hybrid cloud without using levels.

A. Comparison of Hybrid and Public Clouds

When we execute this application on cloudsim simulator which is using proposed level based scheduling algorithm, some tasks are executed on private cloud and some are executed on public cloud. In hybrid cloud, cost is counted only for those tasks that run on public cloud as private cloud resources are considered as free to use resources for an organization. After running workflow applicaton on hybrid cloud, cost and makespan are shown in Fig.2 and Fig.3.

.

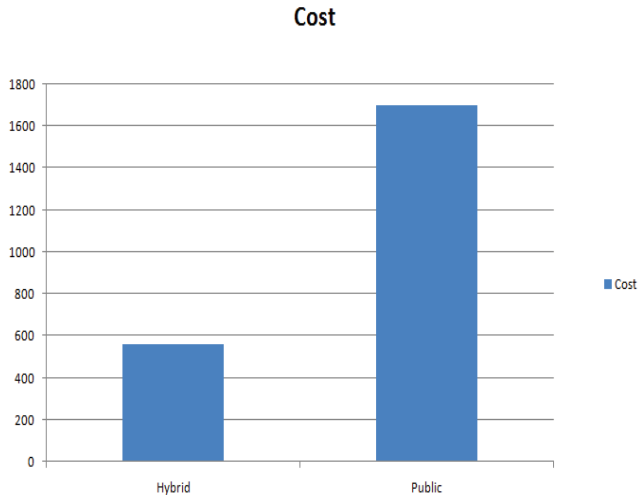

Fig.2.Cost variation in Hybrid and Public clouds

Cost is set for each public machine based on its processing capability. Cost of executing job on public cloud resource is calculated on the basis of time for which the resources are being utilized. Makespan is much lower than the set deadline in both scenarios. On running workflow jobs on public cloud, cost and makespan variations have been shown in Fig.2 and Fig.3.

Public cloud has high configuration hardware so running a job using public cloud resources will definitely reduce the makespan of the application but the cost will increase whereas cost would be less if hybrid cloud is used as private cloud resources are freely available to user in hybrid cloud environment. Cost in hybrid cloud is the cost incurred for the time for which the public cloud resources are being utilized.

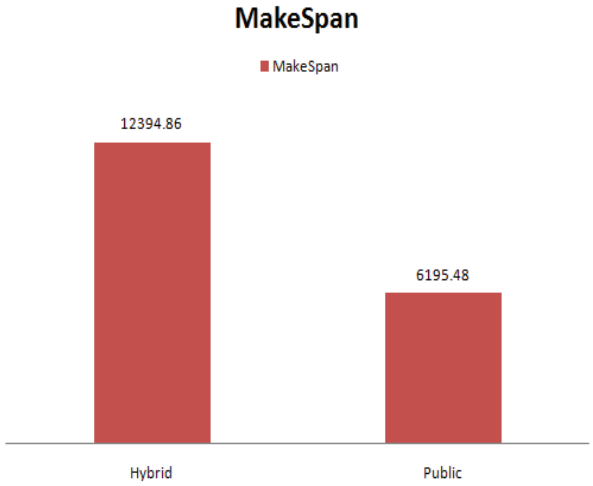

Fig.3.Makespan variation of Hybrid and Public cloud

On analyzing the results we can see that makespan of the workflow job on public cloud is almost half than the makespan of job on hybrid cloud. But cost has increased almost three times. Makespan of workflow application on hybrid cloud is much lower than the set deadline and cost of executing workflow application is also very low on hybrid cloud as compare to public cloud so using hybrid cloud is a better choice.

B. Comparison of level based, without level based and min-min scheduling approach in hybrid cloud

Proposed scheduling algorithm is level based scheduling. In this algorithm whole work flow application is divided into various levels. All the tasks that will start execution in the beginning of the workflow application will be in the first level. Children or successors of the current level tasks will in the next level. Instead of scheduling all tasks of the workflow application at one time, proposed algorithm pick one level tasks at one time and schedule them. When current level tasks finish their execution, their occupied resources will be free to be rescheduled at next level. This approach allows better scheduling of resources at private and public clouds. We can allocate more resources in private cloud by using level based scheduling.

In without level scheduling all tasks of the application will be scheduled on the available resources at once so resources occupied by parents won't be available for the child nodes. Results shows that this approach increases the cost of running the cloud on hybrid cloud and difference in makespan is also not very large than level based scheduling. Fig.4. shows the makespan of level based approach and without level based approach, it also shows the deadline within which the required output of workflow must be achieved.

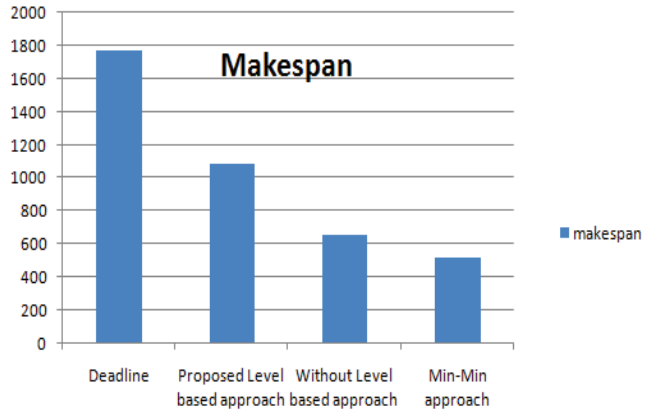

Fig.4.Deadline and Makespan variation in level based, without level based and min-min approach

From the results we can see that workflow application execution has been completed much earlier than the set deadline in all scheduling algorithms. For its performance analysis, comparison of level based algorithm is also done with min-min approach. In min-min approach, tasks which requires minimum execution time is chosen first and the resource which gives minimum estimated execution time is allocated to it. If the private cloud resources are not sufficient to execute workflow application then public cloud resources are taken and the same min-min approach is followed to allocate public cloud resources.

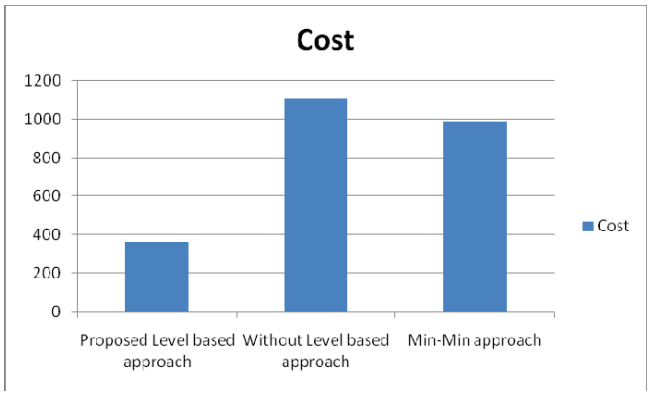

Fig.5. Cost variation in level based, min-min and without level based approach

In fig.4, we can see that makespan is higher in proposed level based algorithm then min-min approach but both approaches completes execution within the workflow deadline which is the requirement of user . In fact, a lot cost is being saved if proposed level based algorithm is used instead of min-min as shown in fig.5. As shown in Fig.4. and Fig.5. for comparison of level based approach with without level based approach, makespan is high in level based but both complete the workflow execution within deadline whereas cost for

execution of workflow is less than 3 times in using level based approach as compared to without level based approach. In comparison of level based approach with min-min approach, makespan of min-min is quite low as compared to level based approach but the requirement of user is to get the required output of workflow before the deadline which is fulfilled by both these approaches. The main concern is cost incurred to execute the workflow. Around 2.7 times cost is saved if proposed level based approach is used instead of min-min. The proposed algorithm outperforms the min-min approach by large extent in terms of cost.

VII. Conclusion

When we compare execution cost of these scheduling algorithms, we can easily notice that level based hybrid scheduling algorithm is doing much better than without level based scheduling and min-min. Execution cost is near about 3 times higher than level based scheduling in without level scheduling algorithm whereas makespan of level based scheduling is only 1.55 times higher than without level scheduling algorithm. This has become possible by introducing the concept of individual task deadline known as sub-deadline of workflow. Moreover when it is compared with min-min it shows that makespan of proposed level based algorithm is almost double than min-min makespan but the cost incurred is around 3 times lesser then the cost incurred in min-min. This makes it essential to use level based scheduling approach to save plenty of cost by maintaining deadlines. In future we are planning to introduce the idea of job grouping as presented in [14] in the current work. We are also planning to work on full workflow application to find the variations of cost and other measures in real time.

References

[1] Zhang, Qi, Lu Cheng, and Raouf Boutaba. "Cloud computing: state-of-the-art and research challenges." Journal of Internet Services and Applications 1, no. 1 (2010): 7-18.

[2] Luiz Fernando Bittencourt, Edmundo Roberto Mauro Madeira, HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds, vol.8. The Brazilian Computer Society 2011, Springer, pp.68–73.

[3] Vouk, Mladen A. "Cloud computing–issues, research and implementations." Journal of Computing and Information Technology 16.4 (2004): 235-246

[4] Goldberg, Robert P. "Architecture of virtual machines." In Proceedings of the workshop on virtual computer systems, pp. 74-112. ACM, 1973.

[5] Abels, Tim, Puneet Dhawan, and Balasubramanian Chandrasekaran. "An overview of xen virtualization." Dell Power Solutions 8 (2005): 109-111.

[6] Mizan, Tasquia, Shah Murtaza Rashid Al Masud, and Rohaya Latip. "Modified Bees Life Algorithm for Job Scheduling in Hybrid Cloud." (2012).

[7] Van den Bossche, Ruben, Kurt Vanmechelen, and Jan Broeckhove. "Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads." In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pp. 228-235. IEEE, 2010.

[8] Yu, Jia, and Rajkumar Buyya. "Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms" Scientific Programming 14, no. 3 (2006): 217-230.

[9] Yu, Jia, Rajkumar Buyya, and Chen Khong Tham. "Cost-based scheduling of scientific workflow applications on utility grids."In e-Science and Grid Computing, 2005. First International Conference on pp. 8-pp. IEEE, 2005.

[10] Sakellariou, Rizos, and Henan Zhao. "A hybrid heuristic for DAG scheduling on heterogeneous systems." In Parallel and Distributed Processing Symposium, 2004. Proceedings 18th International,p.111.IEEE,2004.

[11] Bittencourt, Luiz F., Edmundo RM Madeira, F. R. L. Cicerre, and L. E Buzato. "A path clustering heuristic for scheduling task graphs onto a grid." In 3rd International Workshop on Middleware for Grid Computing (MGC05). 2005.

[12] Lovejit Singh, Sarbjeet Singh, "A Survey of Workflow Scheduling Algorithms and Research Issues", International Journal of Computer Applications (IJCA), Vol. 74, No. 15, July 2013, pp. 21-28.

[13] Calheiros, Rodrigo N., Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." Software: Practice and Experience 41, no. 1 (2011): 23-50.

[14] Simrat Kaur, Sarbjeet Singh, "Comparative Analysis of Job Grouping based Scheduling Strategies in Grid Computing", International Journal of Computer Applications (IJCA), Vol. 43, No. 15, April 2012, pp. 28-35.