Resource Management in the Multi-Tenant Cloud Environment

Piwal Priya M.
PG Student
SSVPS's BS Deore College of Engineering,
Dhule (MH), 424005,
India

Mandre B. R.
Associate Professor and Head
SSVPS's BS Deore College of Engineering,
Dhule (MH), 424005,
India

ABSTRACT
The cloud computing is an Internet-based computing emerging as a new architecture which aims to give reliable, customizable and QoS guaranteed dynamic environment for end-users. As multi-tenancy is one of the key features of cloud computing where service providers and users have scalable and economic benefits on same cloud platforms. In cloud computing environment the execution process requires resource management due to the processing capability is high to the resource ratio. The aim of the system is to handle resource management by executing scientific workflows. The locating and assigning of free resources is handled through the Cloud-based Workflow Scheduling Algorithm (CWSA) policy. The simulation results shows that the scheduling algorithm improves the performance of scientific workflows and helps in minimization of workflow completion time, tardiness, execution cost and use of idle resources of cloud using simulator Workflowsim.

General Terms
Cloud Computing, Workflows, Workflow Scheduling.

Keywords
Cloud Computing, Multi-Tenant Cloud Environment, Scientific Workflows, Resource management.

1. INTRODUCTION
Cloud computing is a concept that is used to describe a paradigm for delivery of computing services to users on a pay-as-you-go basis. The cloud technology permits more efficient computing by removing most of the upfront costs of setting up an IT infrastructure. It also allows organizations to expand or minimize their computing facilities rapidly [2].

Cloud computing often uses the multi-tenancy technique where tenants share same system software. The multi-tenancy, within the software architecture, is referred to as the ability to serve multiple client organizations by one instance of a software application and can be seen as a high level architectural structure in which a single instance of a software application is hosted on the software vendor's infrastructure, and multiple tenants access the same instance [3].

Figure 1 shows the overview of multi-tenancy cloud architecture which has three layers where the architecture works. The first layer of the multi-tenancy cloud is the data center layer that provides the facility of datacenter space, servers, routers, etc. that helps multiple tenant software request. Next layer is the infrastructure layer that accepts through software stack, where single stack is associates to the specific customers. The third layer is the cloud application service layer in which implementing multi-tenancy requires execution at both the software layer and the infrastructure layer [4].

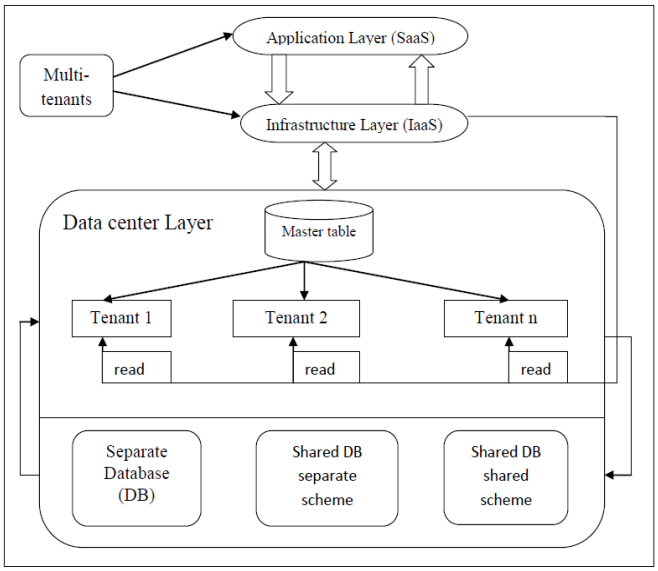

Fig 1: Overview of Multi-Tenant architecture.

Cloud workflow applications (e.g. high-end scalable media processing, scientific workflow applications, and data analytics) can be described as a collection of resource intensive activities processed in order to achieve a goal, which could be executed in geographically distributed heterogeneous resources. A cloud workflow can be modeled as a Directed Acyclic Graph (DAG). It consists of computational costs and transmission tasks [1].The cloud workflow can be categorized into two groups: balanced structured and unbalanced structured workflows.

Most of the workflow scheduling is deal with the idea of scheduling performance of a single workflow. Moreover, these techniques did not consider compute-intensive workflow applications in a multi-tenant cloud environment. Unlike, our work focuses on the minimization of the makespan, execution cost of the workflows and tardiness while maximizing the resource utilization within a deadline [1].

2. RELATED WORK
In 2004 R. Sakellariou et al developed a novel heuristic algorithm for DAG scheduling on heterogeneous machines, which compares favorably with other related heuristics and shows less sensitivity to different approaches for weighting nodes /edges [5]. In 2012 H. M. Fard et al focused on multi-objective static scheduling of scientific workflows in heterogeneous environment which is list scheduling heuristic proposed for scheduling workflow in grids and clouds [6].

In 2009, W N Chen develops an ant colony optimization approach to schedule workflow applications. It finds the cost of execution of workflow application [7]. In 2010 S Pandey et

al proposed PSO algorithm that allocates resources to workflow applications. It is used for workflow application by varying its computation cost and communication cost [9].]. In 2013 H M Fard et al introduces a pricing model and a truthful mechanism for scheduling single tasks considering: monetary cost and completion time [10].

In 2009 G Juve et al introduces some of the ways in which data can be managed for workflows in the cloud. In grids and clusters, workflow data is often stored on network and parallel file systems [11]. A Two-tier SaaS scaling and scheduling architecture that works at both service level and application level to save resources, and the idea is to increase the resources to those components only was described in [12].

3. SYSTEM MODEL

Scientific workflows are used to compose and execute computational tasks that are control and data dependent. The performance of scientific workflow is high in throughput computing and data analysis. Scientific workflow applications have a discrete structure and requires heterogeneous environment. The services executed by such workflow application faces challenges such as scalability, quality of service (QoS), reproducibility, data storage, etc. The workflow structure represents the sequence of execution of tasks. Based on the workflow representation the cloud workflow can be modeled as a Directed Acyclic Graph (DAG) which consists of transmission and computation tasks. Figure 2 shows the representation of DAG-based workflow. In Figure 2 the tasks A, B, F represents the sequence and the parallelism tasks are B, C, and D.

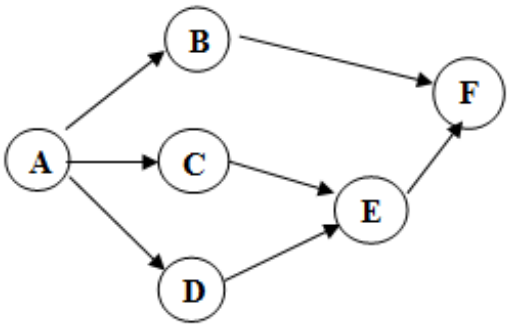

Fig 2: Representation of DAG-based workflow

Initially, the DAG workflow is represented in a XML file format and further processed by tool like Pegasus workflow management system to generate DAG file. The DAG based workflow is a representation of graph $G = (V,E)$, where V is the set of vertices representing tasks and E denotes the set of communication edges representing precedence relation between two tasks. The vertex V is in the form of $V = \{T_i | i = 1, \dots, v\}$ with $|V| = v$ and E is in the form $E = e_{ij} | (i,j) \in \{1, \dots, v\} \times (1, \dots, v\}$ with $|E| = e$. An edge e_{ij} is of form (v_i, v_j) presents is there is dependency between tasks v_i and v_j. The task v_i is said the parent of task v_j and v_j is called as child of task v_i. The child task could not be totally executed until its entire parent tasks are executed. Entry task is a task which had no predecessor task and the exit task is a task having no successor task. Thus DAG has large number of entry and exit task [13].

3.1 System Architecture

Figure 3 depicts the layered architecture proposed workflow scheduling system. First input is given as XML file which represents Directed Acyclic Graph (DAG) structure of workflow to the algorithm by tenant or user. The Directed Acyclic Graph in XML format (DAX) is generated of the workflow applications for given number of tasks. Thus DAX file contains a list of tasks and the dependencies between them and the computation time and input/output data size of each task. Afterwards the information in DAX file is submitted for scheduling in the workflow scheduler which is the core component. The scheduled tasks are optimized by CWSA algorithm for performing workflow scheduling and resource management.

3.2 Cloud-based Workflow Scheduling Algorithm (CWSA)

Initially, the CWSA algorithm takes unscheduled workflows as input with aim to schedule workflow and submit it to the workflow scheduler. The tasks in the workflow would be inserted into service queue. The time period of an idle CPU that is the $schedule\_gap$ is calculated (as in equation 1) to select a better schedule position according to current schedule position. As the workflow scheduler executes different scheduling policies, the $schedule\_gap$ is not zero than the workflow scheduler executes the scheduling policies. If $schedule\_gap == 1$ then workflow is scheduled using FCFS, $schedule\_gap == 2$ the workflow tasks would be scheduled using EASY Backfilling, $schedule\_gaps == 3$ then workflow schedule by MCT algorithm. If the $schedule\_gap$ is zero, than workflow tasks scheduled with CWSA () algorithm.

Algorithm 1: Cloud-based Workflow Scheduling Algorithm
Require: Workflow w is defined, New_Schedule=0
Input: w- unscheduled workflow in the list
for (all $R_k \in Res$) do
{
    Find schedule_gap
    Calculate schedule_gap as in equation 1
}//end for
if (schedule_gap!=0)
{
    Switch (scheduling_policy)
    {
        case 1:
            schedule workflow w = New_Schedule using FCFS on $R_k \in Res$ resources
            break;
        case 2:
            schedule workflow w New_Schedule using MCT on $R_k \in Res$ resources
            break;
        case 3:
            Schedule workflow w New_Schedule using EASY Backfilling on $R_k \in Res$ resources
            break;
    }//end switch
}//end if
else if
{
    CWSA()
}//end else if
Function 1 function CWSA ()
CWSA ()
New_Schedule = Initial_Schedule
if (total_weight>0)
{
    for (all $R_k \in Res$)
    {

New_Schedule <- move workflow w into schedule_gap
}//end for
else
remove workflow w from schedule
}//end if
End function

The $schedule\_gap$ is occurs when CPU is in idle time period. The $schedule\_gap$ is calculated which means a proper time slot of resources for each ready workflow task. The workflow w with deadline $d_l, l = 1,2, \ldots . N$ of a scheduled workflow with schedule utilization $su(t)$ is defined as time t is utilize by the current task invocation, which is calculated as,

$$su(t) = \sum_{ti_l \in ci(t)} (ci/ti) \leq 1 \qquad \text{eq. (1)}$$

The total_weight is the highest value of weight for each assigned of a new workflow task, which is set as ideal CPU gap. The total_weight is calculated as,

$$total_{weight} = weight_{makespan} + weight_{deadline} \quad eq. (2)$$

Where,

$$weight\_makespan = \frac{Makespan\_old - Makespan\_new}{Makespan\_old}$$

$$weight\_deadline = \frac{Nondelayed\_new - Nondelayed\_old}{Nondelayed\_old}$$

4. SIMULATION ON WORKFLOWSIM

WorkflowSim tool is an extension tool of existing CloudSim simulator which provides more layers of workflow management and effective evaluation platform. There are three main components of WorkflowSim: a Workflow Mapper, a Workflow Engine and a Workflow Scheduler. The WorkflowSim depends on CloudSim for providing accurate and suitable task level execution model, such as time-shared and space-shared model. Moreover, WorkflowSim shows various layers of overheads and failures, which improves in accuracy of simulation.

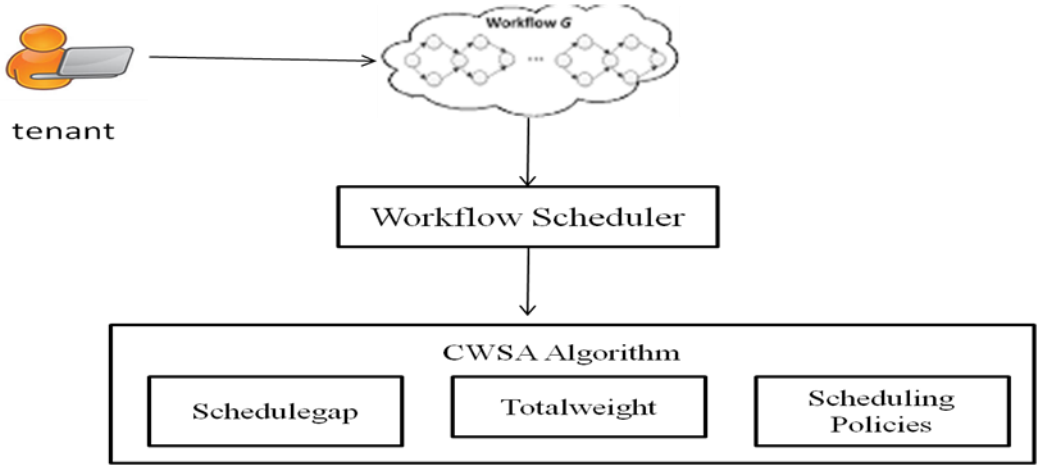

Fig 3: System Architecture

4.1 Simulation Description

Virtual Machine – It is based on computer architecture and provides functionality of a physical computer.

Cloudlet – Cloudlet contains set of tasks that executes in cloud environment. It has its parameters like Cloudlet_ID and Cloudlet_Length.

The results analysis description was conducted on single machine *Hp* PC 2.4 GHz Intel i3 CPU and 2 GB of memory running windows 8 and WorkflowSim. The ten virtual machines with a single data center is created by using WorkflowSim. The XML files of different workflows are given as input to the algorithm. These workflow contains different number of tasks depending on workload size, and provided for scheduling. Result implementation is done based on different parameters like makespan, tardiness, execution time, resource utilization and CDF with comparison to previous algorithms. Table 1 shows the values of CDF of makespan with performance variation for different scheduling policies. Table 2 depicts the resource utilization values of each scheduling policies and percentage in graph. Table 3 calculates and shows the values of average execution time determining scalability of scheduling policies. Table 4 evaluates and shows the values of average tardiness of workflows for different scheduling policies. Table 5 shows 99th percentile distribution of the makespan to represents measures of scheduling policies for Cybershake workflows.

Table 1: CDF for Cybershake workflow having tasks=100

FCFS	CWSA	EASY	MCT
10	30	7	1
19	50	17	17
32	58	24	20
41	78	68	68
76	83	60	60
80	95	68	58

Table 2: Resource utilization of Cybershake workflow with tasks=50

Size	FCFS	EASY	MCT	CWSA
30	90	80	78	83
50	95	80	75	85
100	95	85	75	75
1000	94	85	77	73

Table 3: Average execution time for Cybershake workflow tasks=50

FCFS	CWSA	EASY	MCT
600	500	900	800
800	700	700	600
1600	700	1600	1400
1800	1000	1800	1600

2200	2500	2100	1800
2500	1900	2500	2200
2900	2400	2900	2400
2500	3300	3500	2900

Table 4: Average tardiness of Cybershake workflow with Tasks =100

FCFS	CWSA	EASY	MCT
20	10	40	60
30	10	48	61
50	20	55	70
70	28	70	80
90	30	98	98
110	38	120	110
120	39	130	118

Table 5: 99th percentile of Cybershake workflow application

FCFS	CWSA	EASY	MCT
58	60	43	30
40	55	40	30
36	45	27	17
22	30	21	7
19	18	15	15
10	12	6	5

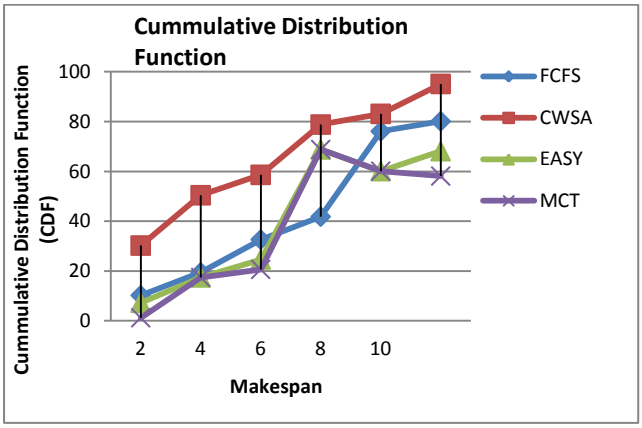

Fig 4: CDF for Cybershake workflow with tasks=100

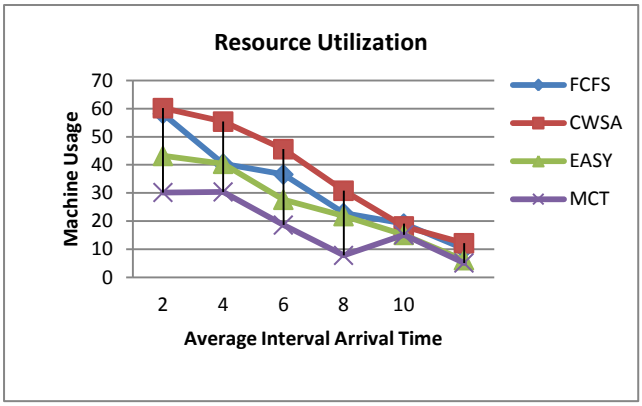

Fig 5: Resource utilization for Cybershake workflow with tasks=50

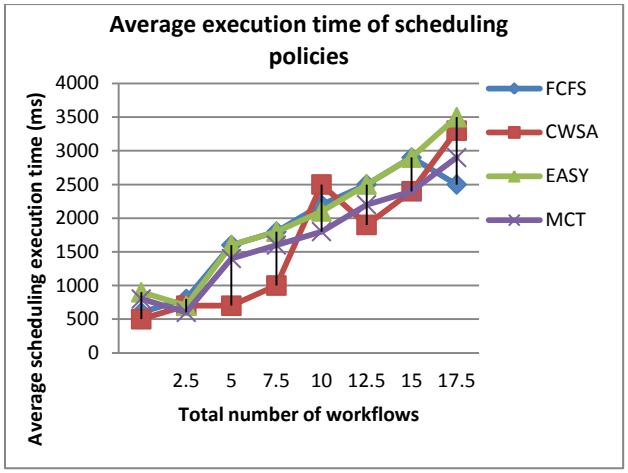

Fig 6: Average execution time for Cybershake workflow with tasks=50

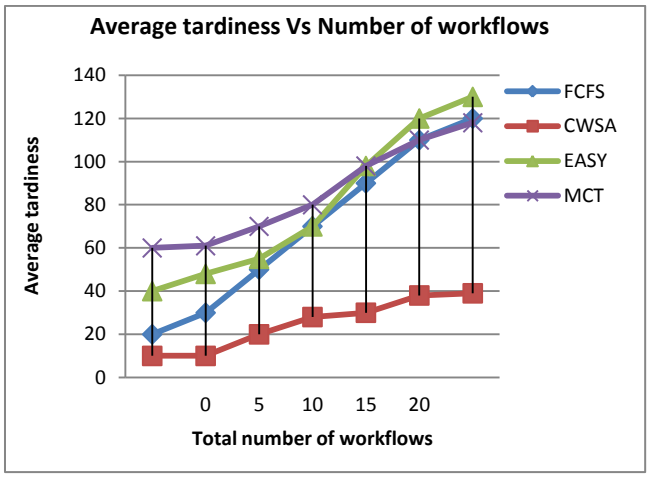

Fig 7: Average tardiness for Cybershake workflow with tasks=50

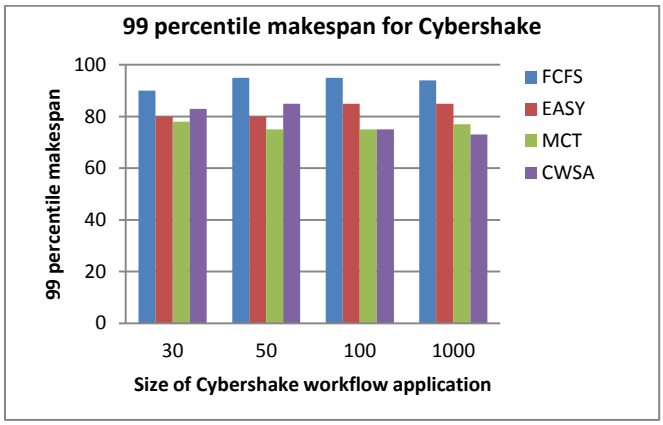

Fig 8: 99th percentile makespan for Cybershake workflow with tasks=100.

5. CONCLUSION

Scheduling is an important process in cloud computing environment as this computing paradigm executes the compute-intensive workflows and scientific workflow applications for executing large number of scientific data. Large number of researches has been extensively studied in workflow scheduling with various objectives to obtain

optimized solutions. Most of the existing scheduling algorithm is focused on single cloud environment and suitable for single workflow. Thus problem of resource management in multi-tenant cloud is due to resource isolation and heterogeneity. This paper represents the CWSA which helps in executing workflows and locate and assign free resources to the virtual machine when they are free in scheduling process as well as utilize idle resources in the system, minimizes makespan, tardiness, and execution cost of workflow which performs better than previous algorithms. For the future work, the present CWSA scheduling policy would be execute in terms of mobile computing.

6. REFERENCES

[1] Martin Maier, Bhaskar Prasad Rimal, "Workflow Scheduling in Multi- Tenant Cloud Computing Environments," *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, no. 99, pp. 1-1, JULY 2015.

[2] R. N. Calheiros, R. Buyya A N Toosi, "Interconnected cloud computing environments: challenges, taxonomy and survey," *ACM Computing Surveys*, vol. 47, no. 1, Apr 2014.

[3] John Grundy, Jacky Keung Jia Ru, "Software Engineering for Multi-tenancy Computing Challenges and Implications," *InnoSWDev'14*, 16 Nov 2014.

[4] Rizos Sakellariou and Henan Zhao, "Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems," in *Parallel and Distributed Processing Symposium IEEE International Conference.*, 2004.

[5] R. Prodan, J. J. D. Barrionuevo, and T. Fahringer, H. M. Fard, "A multi-objective approach for workflow scheduling in heterogeneous environments," in *in Proc., IEEE/ACM CCGrid*, May 2012, pp. 300-309.

[6] Wei Neng Chena and Jun Zhang, "An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem with Various QoS Requirements," *System, Man and Cybernetics, Applications and Reviews IEEE Transactions*, vol. 39, no. 1, pp. 29-43, 2009.

[7] The XML files that describe of the workflow applications are available via the Pegasus project. [Online]. https://confluence.pegasus.isi.edu/display/ pegasus/WorkflowGenerator.

[8] L.Wu, S. Guru, and R. Buyya S. Pandey, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environment," *IEEE Advanced Information Networking and Applications* , pp. 400-407, Apr 2010.

[9] R. Prodan, and T. Fahringer H. M. Fard, "A truthful dynamiv workflow scheduling mechanism for commercial multicloud environments," *IEEE Trans Parallel and Distrib. Syst.*, vol. 24, no. 6, pp. 1203-1212, June 2013.

[10] E. Deelman, K. Vahi, G. Mehta, B. Berriman, B. Berman and P. Maechling G. Juve, "Scientific workflow applications on amazon ec2," in *IEEE E-Science Wksp*, 2009, pp. 59-66.

[11] X. Sun, Q. Shao, and G. Qi W. Tsai, "Two-tier multi-tenancyscaling and load balancing," *in Proc., IEEE ICEBE*, vol. 9, no. 1, pp. 484-489, Nov 2010.

[12] B.Latha, AND G. Sumathi D.A.Prathibha, "Efficient Scheduling of Workflow In Cloud Enviornment using Billing Model Aware Task Clustering," *Journal of Theoretical and Applied Information Technology*, vol. 65, no. 3, pp. 595-605, july 2014.

[13] B.Latha, AND G. Sumathi D.A.Prathibha, "Efficient Scheduling of Workflow In Cloud Enviornment using Billing Model Aware Task Clustering," *Journal of Theoretical and Applied Information Technology*, vol. 65, no. 3, pp. 595-605, july 2014.