International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 12 (2017) pp. 3311-3317
© Research India Publications. http://www.ripublication.com

Q Learning Based Workflow Scheduling in Hadoop

Rashmi S[1]
Department of Computer Science and Engineering,
East Point College of Engineering and Technology, Bangalore, India.

Dr Anirban Basu
Department of Computer Science and Engineering,
Acharya Pathasala College of Engineering, Bangalore, India.

[1]***ORCID: 0000-0002-6966-5647***

Abstract

Hadoop on datacenter is the popular analytical platform for enterprises. Cloud vendors host Hadoop clusters on the datacenter to provide high performance analytical computing facilities to its customers. While many concurrent users try to use the Clusters to execute their jobs, scheduling should be very effective to complete their job in time and at same time use the resources efficiently with effective cost and time management. Workflows are repeatable pattern of dependable jobs. The workflows are executed in the Hadoop datacenter by allocating VMs. In our earlier papers, a mechanism to pack and execute the customer jobs as workflows on Hadoop platform was proposed which minimizes the VM cost and also executes the workflow Hadoop-MapReduce jobs within deadline.In this paper, we propose a Q learning based scheduling method to optimize the cloud resources in workflows. Q Learning is a model free reinforcement learning technique used to find an optimal action – selection policy for a given Markov decision process. The parameters considered for optimization are VMs consumed, bandwidth at data center and the electric power consumption.

Keywords: Hadoop, Workflow Scheduling, Q-Learning, Markov decision process

INTRODUCTION

Large scale analytics in Datacenter is a most important requirement in modern business environment to process information effectively. Datacenters employ MapReduce paradigm [1] based on Hadoop [2] to provide large scale data and compute intensive analytics. As the users submit jobs, Hadoop clusters in the datacenter executes them on separate VMs and returns results.

Workflows [3] are a set of jobs with dependencies between them. Workflow as a whole has a deadline execution time. In a Workflow, some VMs need to be maintained even after the execution of jobs is completed; either because other jobs are dependent on these jobs or their results has to be combined.

Scheduling workflows on VMs in datacenter must be done effectively to use the resources in a best possible manner and must also be adapted for this new requirement. The resources considered are the VM consumed bandwidth at datacenter and electric power consumption. While optimizing these parameters, the deadline of workflows must not be compromised.

Reinforcement Learning is a machine learning technique that tries to maximize performance by assisting a software agent to find out just the right behavior automatically in some specific circumstance.

Q-learning is one of the approaches towards model-free reinforcement learning technique. In particular, the goal of Q-learning is to find an optimal action-selection policy for a given Markov decision process (MDP). It attempts to learn an action-value function using feedback from its environment. Moreover, Q-Learning uses temporal differences to estimate the value of this Q-function. The main benefits of using Q-Learning are that, though Simple and dynamic, does not require modeling the environment.

In this paper, we propose a Q learning based approach for resource optimization in Hadoop datacenter for workflows. A datacenter with multiple sub networks and switches connecting these sub networks is considered for designing the proposed solution. Also, network topology is taken into account in view of the fact that effective bandwidth optimization cannot be done with a flat network as considered by most of the resource optimization methods.

RELATED WORK

In this section we survey the current schedulers in datacenter and identify their shortcoming for workflows.

A business application in the modern cloud era is essentially reliant on large scale data and compute intensive applications. In such applications, data and computation – intensive jobs are depicted as a workflow or dataflow. Parallel Programming model such as MapReduce can be used to execute Map and Reduce jobs concurrently to improve execution efficiency.

The growth rate of data in these applications is exponential which requires the execution rate and resource availability also to rise in the same pace. Hence there is a need for an efficient Scheduling algorithm that enhances performance and improves resource utilization in Hadoop MapReduce clusters. Though schedulers such as FIFO, Fair Scheduling, Capacity scheduling [4] [5] etc., are quite successful, there is still room for improvement. Improved Schedulers such as COSHH [6], DREAMS [7], PRISM [8], and Resource Stealing mechanisms [9] are also available. But each comes with its own pros and cons [10].

Apache Oozie [11] is a popular workflow Scheduler for Hadoop jobs. It ponders over characteristics such as Multi-tenancy, Operability, Scalability and Security. Cascading [12] is a platform to develop complex workflow based on data processing applications and deploy it onto the Hadoop clusters. NOVA [13] is yet other method which acts as workflow manager used by Yahoo. PFAS [14] is proposed for grids. It concentrates on reducing the overall completion time of the tasks in the workflow. It also focuses on improving resource utilization by allocating unscheduled tasks beforehand. ΦSched [15] is a hardware heterogeneity aware scheduler for workflows with MapReduce jobs. The scheduling approach leverages on data placement and data locality to decrease the execution time and to increase the throughput. SAMR [16] is a speculative scheduling algorithm which tries to improve on resource utilization by conserving resources and initiate a back up of the slow tasks only if it does not decelerate the system. Hadoop cluster scheduling is proposed for Kepler-Hadoop map reduce scientific workflows [17]. It uses a data model that piles up the provenance data for the MapReduce workflows.

D. de Oliveira et al [18] proposed a provenance-based adaptive scheduling for scientific workflows in clouds. Here, the scheduling is based on the total execution time, cost and reliability. The costing is based on VM selection. It shows an improved performance in scenarios where the number of VMs remain consistent.

D. de Oliveira et al [19] also proposed adaptive parallel execution strategy for workflows. The solution relies on adaptability with respect to resource requirement and computation capability in a varying environment. K. Deng et al [20] have proposed another scheduling strategy for scientific workflows. This method considers three categories of dependencies (data-task, data-data and task-task) in order to balance the load.

According to our survey, not many proposals have been put forward to schedule map and reduce tasks based workflows on the cloud with deadline constraints and effective resource cost.

PROBLEM DEFINITION

Clouds have their roots from the profound concept of virtualization and virtual machines play a significant role. Once map reduce jobs are provided to the datacenter in the form of workflows, the scheduler must plan and allocate the VMs to these jobs in such a way that the workflows are completed within the specified deadline with optimal usage of VMs , bandwidth and energy cost.

A Workflow can be represented as W={S, T, D, d} [21] where S = {J_1, J_2, J_3….J_n} J_1, J_2, J_3….J_n are all Hadoop map/reduce jobs.T= {T_1, T2, T3… T_n} T1, T2, T3… T_n is the expected number of transactions to be processed in jobs J_1, J_2, J_3….J_n . D = {J_1->J_2, J_2->J_3 …} represents the dependency constraint among the jobs. J_1->J_2 says that Job J_2 is dependent on the outcome of J_1. d represents the deadline time for the completion of the workflow .

If the scheduler provides some solution ‘X’ to schedule jobs to VM at cost ‘CX’ , then the solution ‘X’ is said to be an optimal solution only if there is no other solution ‘Y’ that executes at cost ‘CY’ with deadline d such that CY<CX.

We try to optimize three parameters: numbers of VMs, energy cost, bandwidth cost and at the same time execute the workflow within deadline.

The network traffic matrix is represented in form of

$$A_{\alpha\beta} = \begin{bmatrix} A11 & A12 & . & A1n \\ A21 & A22 & . & A2n \\ . & . & . & . \\ Am1 & Am2 & . & Amn \end{bmatrix}$$

Where A12 is the number of packets transmitted per second from VM 1 to VM2.

The network topology matrix is represented in form of

$$H_{\alpha\beta} = \begin{bmatrix} H11 & H12 & . & H1y \\ H21 & H22 & . & H2y \\ . & . & . & . \\ Hx1 & Hx2 & . & Hxy \end{bmatrix}$$

Where H12 is the number of switches hop from VM 1 to VM2.

Q LEARNING BASED SCHEDULING

The architecture of the proposed solution

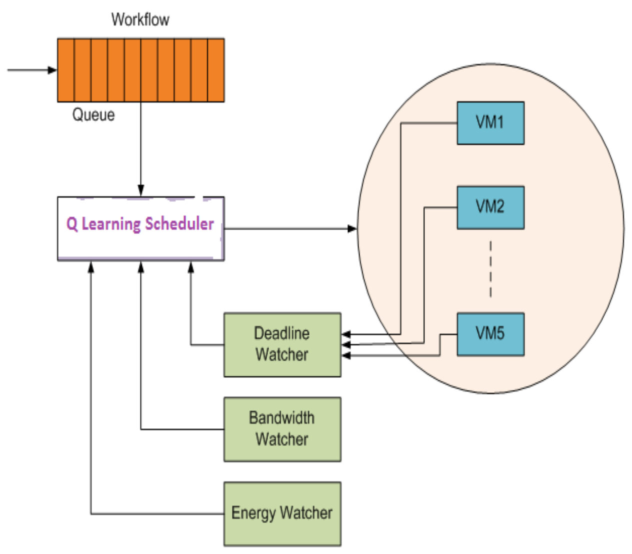


Figure 1: Architecture for Scheduler based on Q Learning Method

Workflow are queued and processed in FIFO order.

The scheduling consists of following steps

1. Job Preprocessing
2. Q Learning Scheduling

1. Job Preprocessing:

Once the workflow W={S, T, D, d} is given and the VM capability VC= {V_1,V_2… V_n} ,the network topology and the network traffic matrix are available , the scheduling starts.

As each VM is of different capability, the average configuration can be considered as a reference. By traversing backwards with the given workflow deadline d from the final task, the earliest start time and the deadline for each task in the workflow is calculated. The details of the Task, its earliest start time and the deadline time are added to the Task time Queue.

A Timed queue can be used to save the tasks with same earliest start time in a linked list in the same position.

2. Q Learning Scheduling:

The task with next earliest start time are taken from timed queue and each task is scheduled to VM according to the Q Learning Procedure.

Q-learning [22] is one of the approaches towards model-free reinforcement learning technique. The goal of Q-learning is to find an optimal action-selection policy for a given Markov decision process (MDP). It attempts to learn an action-value function using feedback from its environment

The main components of Q-Learning algorithm are a set of states S , a software agent , a set of actions A and a set of rewards R. The software agent tries to learn an optimal policy to maximize its rewards by performing an action a € A in a particular state.

The algorithm therefore has a function that calculates the Quantity of a state-action combination:

$$\mathbf{Q : S \times A \rightarrow R}$$

The agent refers to a sequence of state-action-rewards:

$$\langle s_0,a_0,r_1,s_1,a_1,r_2,s_2,a_2,r_3,s_3,a_3,r_4,s_4\ldots\rangle,$$

which means that the agent was in state s0 and did action a0, which resulted in it receiving reward r1 and being in state s1; then it did action a1, received reward r2, and ended up in state s2; then it did action a2, received reward r3, and ended up in state s3; and so on.

This can be summarized as an experience in which a tuple $\langle s,a,r,s'\rangle$, means that the agent was in state s, it did action a, it received reward r, and it went into state s'. The tuple $\langle s,a,r,s'\rangle$ provides a datapoint for the Q function $Q(s,a)$. The data point in turn provides a new data point called return which is the sum of actual current reward and discounted estimated future value. Q-learning uses temporal differences to estimate the value of $Q^*(s,a)$.

$$\underbrace{Q[s,a]}_{1} \leftarrow Q[s,a] + \alpha\,(r + \gamma\, \underbrace{\max_{a'} Q[s',a']}_{2} - \underbrace{Q[s,a]}_{3})$$

where

α is the learning rate, set between 0 and 1

γ - discount factor, set between 0 and 1

max $_\alpha$ - the maximum reward that is attainable in the state following the current one. i.e the reward for taking the optimal action thereafter.

Q[s,a] - enclosed in braces 1 and 3 are the old values of Q

$max_{a'}\,Q[s',a']$ – enclosed in braces 2 is the estimate of optimal future value.

The pseudo code[23] of Q – Learning algorithm is given as

```
Inputs
        S is a set of states
        A is a set of actins
        $\gamma$ the discount
        $\alpha$ is the step size
Local
   real array Q[S, A]
   previous state
   previous action a
   initialize Q[S, A] arbitrarily
   observe current state s
repeat
   select and carry out an action a
   observe reward r and state s′
  $Q[s, a] \leftarrow Q[s, a] + \alpha (r + \gamma \max_{a'} Q[s', a'] - Q[s, a])$
   $s \leftarrow s'$
until termination
```

The cost of allocation of task is modeled as

Cost = Cost of VM creation + Cost of Data transfer + Energy increment cost on PM

For the problem of VM allocation and scheduling, the cost increment is split to four intervals and corresponding states are designed S1, S2, S3, S4 and each range is allocated to a state.S1 is state of very low cost increment; S4 is the state of very high increment. The cost increment ranges from -100 to +100. Negative increment refers to punishment in terms of deduction of cost increment and positive increment refers to reward 'r'. The action 'a' refers to allocation of VMs..

The reward for state transitions is given as

Table 1: State transition table

States	**S1**	**S2**	**S3**	**S4**
S1	100	-30	-60	--90
S2	60	30	-30	-60
S3	60	30	10	-60
S4	100	90	60	30

The first VM is allocated to a PM (Physical Machine) with enough capacity and the system goes to state S1 . From this state, next VM with earliest start time is taken from job queue and processed.

For all combinations of allocation of VMs to the physical machine, the cost increment is calculated. The best possible schedule within constraints is choosen for allocation and the corresponding state is migrated to maximize the reward.

The model for VM cost, Energy cost, Bandwidth cost can be derived. VM cost is defined the sum of VM setup cost, VM migration cost , VM running cost.

VMC = VMS + VMM + VMR where

VMC = total VM cost

VMS = VM setup cost

VMM = VM migration cost

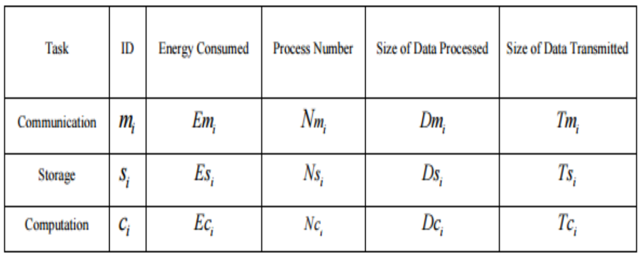
VMR = VM running cost.

If N VM's are created and NM VM's are migrated and each VM runs for time VMT , then unit cost for setup, unit cost for maintenance and unit cost for running are PS, PM and PR respectively.

$$VMC = N*PS + NM*PM + \sum_{i=1}^{N} VMT\hat{}*PR$$

The cost is thus dependent on the number of VMs created. If the number of VMs created can be controlled, setup cost and maintenance cost can be saved.

Energy cost is defined as sum of energy cost of each individual hosts and switches. Energy is modeled in terms of energy consumed for communication, storage and computation.



Task	ID	Energy Consumed	Process Number	Size of Data Processed	Size of Data Transmitted
Communication	m_i	Em_i	Nm_i	Dm_i	Tm_i
Storage	s_i	Es_i	Ns_i	Ds_i	Ts_i
Computation	c_i	Ec_i	Nc_i	Dc_i	Tc_i

Figure 2: Parameters for cost in cloud

$$Em_i = fm_i(Nm_i, Dm_i, Tm_i, SC_i)$$

$$Es_i = fs_i(Ns_i, Ds_i, Ts_i, SC_i)$$

$$Ec_i = fc_i(Nc_i, Dc_i, Tc_i, SC_i)$$

The energy consumed by cloud tasks E_{Dyn} is formulated as follows:

$$E_{Dyn} = \sum_{i=1}^{n} Em_i + \sum_{i=1}^{i} Es_i + \sum_{i=1}^{j} Ec_i$$

Bandwidth cost is defined as the sum of bandwidth used at each switch. Say there are M switches and Di is the outgoing bytes per second in each switches, the bandwidth is modeled as

$$\text{BC} = \sum_{i=1}^{N} Di * T$$

The proposed solution tries to minimize the cost as much as possible.

$$\text{CT} = \text{VMC} + \text{EC} + \text{BC}$$

Four states S1, S2, S3, and S4 are assumed in our work. The cost increment in states is

$$S1 > S2 > S3 > S4$$

The reward for migration of lower to higher is configured in negative increment and the higher to lower is configured in positive increment.The increment or decrement value is proportionate to the level of jump.

Increment (S4→S1) > Increment (S2→S1) and

Decrement (S1→S4) <Decrement (S3→S4)

As Q-Learning tries to optimize the reward , then the number of low state to high state transitions is always preferable than higher to low state transitions

$$L1 = \sum_{N} Si - Sj \quad ¥\, i < j$$

$$L2 = \sum_{M} Si - Sj \quad ¥\, i > j$$

For T = M+ N, is the total number of transitions and L2> L1, As L2>L1, the total cost of scheduling CT is optimized.

RESULTS

The proposed solution was implemented in cloudsim environment. We tested the solution with different workflow for various deadline. We used proposed solution in [21] and [22] for comparison of our scheme.

We measured following parameters

1. Average Resource Utilization
2. VM cost
3. Energy cost
4. Bandwidth cost
5. SLA violation

Jobs with different deadlines are provided to the system and the above mentioned parameters are measured. All jobs are set to transfer 1 Mb of data to next job in the chain.

Average resource utilization is measured by the average CPU utilization of the all the physical machines in the cloud setup. We varied the number of jobs and the result is shown below

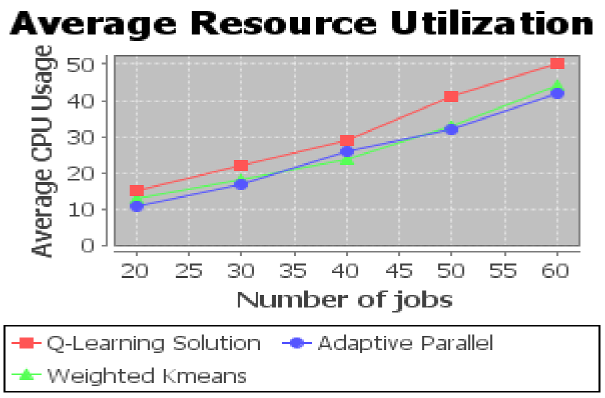


Figure 3: CPU Usage Vs Number of Jobs graph

From the result, we see that the proposed Q-Learning Scheduling has higher average CPU utilization compared to [19] and [20].

VM cost is measured by addition of VM running cost, VM startup cost, VM shifting cost. By varying the number of jobs, VM cost is measured and the result is below

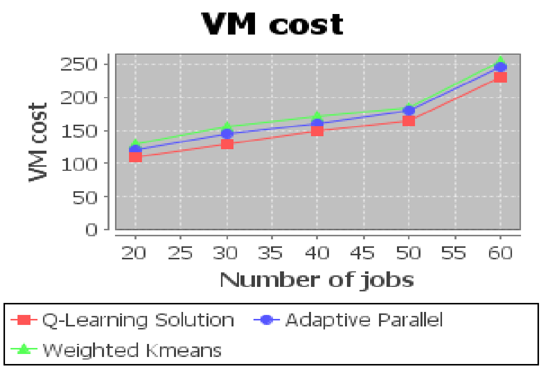


Figure 4: VM Cost Vs Number of Jobs graph

VM cost in the proposed Q-Learning Scheduling is lower compared to other solutions, because of the VM reuse behavior.

Energy cost is measured in terms of total power consumption (watts) of the data center by varying the number of jobs. The result is below

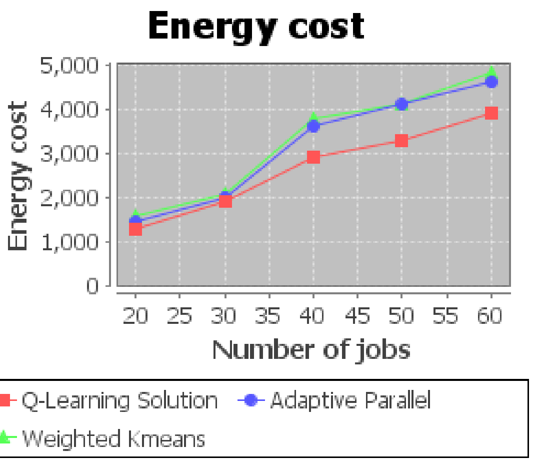


Figure 5: Energy Cost Vs Number of Jobs graph

From this graph, we see that energy consumption in proposed Q-Learning Scheduling is less compared to [19] and [20].

Bandwidth cost is measured in terms total network consumption(Kb) measured at all the switches by varying the number of jobs.

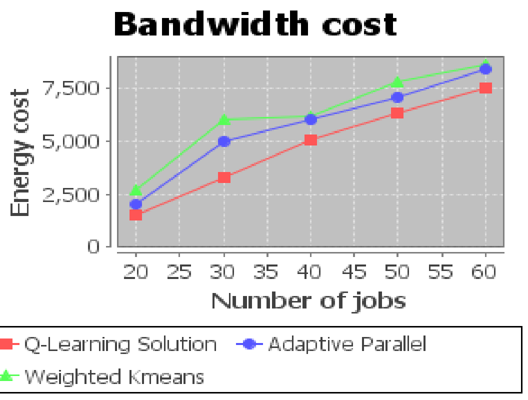


Figure 6: Bandwidth Cost Vs Number of Jobs graph

The bandwidth cost in the proposed Q-Learning Scheduling is less compared to other solutions because of the locality factor considered in VM allocation. The VM with more communication is allocated on the same switch to reduce the amount of data transfer.

SLA violation is measured in terms of percentage of tasks missing the deadline. We varied the number of task and measured the SLA violation and the result is below

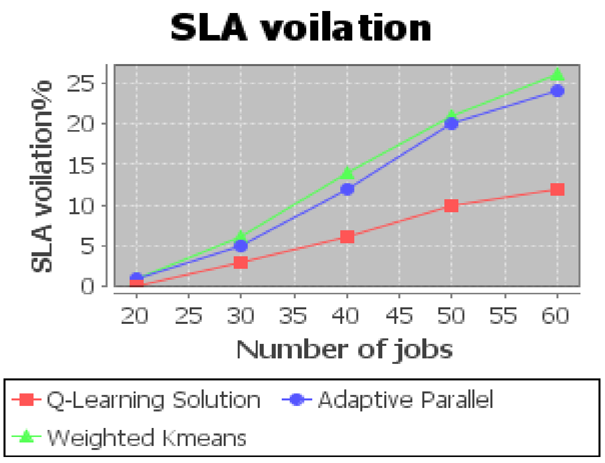


Figure 7: SLA Violation Vs Number of Jobs graph

From the result we see that the SLA violation is lower in the proposed Q-Learning Scheduling compared to [19] and [20].

CONCLUSION

The proposed Q-Learning based scheduler is able to execute the jobs within their deadline and also optimize parameters such as VM cost, bandwidth cost and the energy cost. The Q-Learning Scheduling has a learning threshold, after which an increase in performance is noticed. This is the time needed for stabilization of learning algorithm to learn the rules for optimized cost. In this paper, the parameters considered for cost optimization are given equal importance but in future we plan to use varied weights for each factor and measure the performance.

REFERENCES

[1] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. Commun. ACM, 51(1):107–113, 2008.

[2] Hadoop: The Definitive Guide, Second Edition, by Tom White, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472

[3] Li, S. Hu, S. Wang, L. Su, T. Abdelzaher, I. Gupta, and R. Pace, "WOHA: Deadline-Aware Map-Reduce Workflow Scheduling Framework over Hadoop Clusters," in International Conference on Distributed Computing Systems, pp. 93–103, 2014.

[4] Hadoop'sFair cheduler.https://hadoop.apache.org/ docs/ r1.2.1/fair_scheduler

[5] Hadoop'sCapacitySchedulerhttp://hadoop.apache.org/ core/docs/current/capacity_scheduler.html

[6] Aysan Rasooli , Douglas G. Down, "COSHH: A Classification and Optimization based Scheduler for Heterogeneous Hadoop Systems", in July 2014

[7] Z Liu, Zhang, Zhani, Boutaba,Y Liu, Gong," DREAMS: Dynamic Resource Allocation for MapReduce with Data Skew", 2015

[8] Qi Zhang, Mohamed Faten Zhani, Yuke Yang, Raouf Boutaba,and Bernard Wong," PRISM: Fine-Grained Resource-Aware Scheduling for MapReduce", in Jun 2015

[9] Zhenhua Guo, Geoffrey Fox , " Improving MapReduce Performance in Heterogeneous Network Environments and Resource Utilization", in 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2012

[10] S. Rashmi , AnirbanBasu," Scheduling Strategies in Hadoop: A Survey", in OJCST,ISSN : 0974- 6471, December 2015,Vol. 8, No. (3), Pgs. 234- 240

[11] M. Islam, A. K. Huang, M. Battisha, M. Chiang, S. Srinivasan, C. Peters, A. Neumann, and A. Abdelnur. Oozie: towards a scalable workflow management system for hadoop. In Proc. ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies, 2012.

[12] C. Wensel. Cascading: Defining and executing complex and fault tolerant data processing workflows on a hadoop cluster, 2008

[13] C. Olston, G. Chiou, L. Chitnis, F. Liu, Y. Han, M. Larsson, A. Neumann, V. B. Rao, V. Sankarasubramanian, S. Seth, et al. Nova: continuous pig/hadoop workflows. In Proc. ACM SIGMOD, 2011.

[14] F. Dong and S. G. Akl. PFAS: a resource-performance-fluctuation-aware workflow scheduling algorithm for grid computing. In Proc. IEEE IPDPS, 2007.

[15] Krish K. R. , Ali Anwar, Ali R. Butt , "ΦSched: A Heterogeneity-Aware Hadoop Workflow Scheduler", in proceedings of 22nd International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems, IEEE, pp. 255-264, Sep. 2014.

[16] Quan Chen ,Daqiang Zhang, Minyi Guo, Qianni Deng,Song Guo, "SAMR: A Self-adaptive MapReduce Scheduling Algorithm In Heterogeneous Environment", in proceedings of 10th IEEE International Conference on CIT, pp. 2736-2743, 2010.

[17] D. Crawl, J. Wang, and I. Altintas. Provenance for mapreduce-based data-intensive workflows. In 6th Workshop on Workflows in Support of Large-scale Science, pages 21–30, 2011.

[18] D. de Oliveira, K. A. C. S. Oca˜na, F. Bai˜ao, and M. Mattoso. A provenance-based adaptive scheduling heuristic for parallel scientific workflows in clouds. Journal of Grid Computing, 10(3):521–552, 2012.

[19] D. de Oliveira, E. Ogasawara, K. Oca˜na, F. Bai˜ao, and M. Mattoso. An adaptive parallel execution strategy for cloud-based scientific workflows. Concurrency and Computation: Practice & Experience, 24(13):1531– 1550, 2012.

[20] K. Deng, L. Kong, J. Song, K. Ren, and D. Yuan. A weighted k-means clustering based co-scheduling strategy towards efficient execution of scientific workflows in collaborative cloud environments. In IEEE 9th Int. Conf. on Dependable, Autonomic and Secure Computing (DASC), pages 547–554, 2011.

[21] Rashmi S and Dr Anirban Basu , " Deadline constrained Cost Effective Workflow Scheduler for Hadoop clusters in Cloud Datacenter", Computational Systems and Information Technology for Sustainable Solution ,R.V College of Engineering, Bengaluru, 6th - 8th October, 2016

[22] Maozu Guo, Yang Liu , J. Malec ,"A new Q-learning algorithm based on the metropolis criterion " , Published in: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) (Volume: 34, Issue: 5, Oct. 2004), Page(s): 2140 – 2143

[23] Arafat Habib, Muhidul Islam Khan," Reinforcement Learning based Autonomic Virtual Machine Management in Clouds ", in 5th International Conference on Informatics, Electronics and Vision (ICIEV) , 2016, Pages:1083 -1088