International Journal of Engineering & Technology, 7 (4) (2018) 4841-4845

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.2702
Research paper

Scheduling Algorithms in Cloud Environments: A Comparative Study

Juliet A Murali[1]*, T Brindha[2]

[1] Research Scholar,Noorul Islam Centre for Higher Education,Tamilnadu
[2]Assistant Professor,Department of Information Technology,
Noorul Islam Centre for Higher Education,Tamilnadu
** E-mail: Julietamurali@gmail.com*

Abstract

Cloud is a distributed environment, having large capacity data centers. It needs parallel processing and task scheduling. Map Reduce is the programming model for processing this big data. Hadoop is a Java-based open source implementation of the Map-Reduce framework. The task scheduling in the MapReduce framework is an optimization problem. This paper describes some advantages, disadvantages, approaches used and the performance metrics comparison of different cloud scheduling algorithms and Hadoop Map Reduce scheduling algorithms.

Keywords: *Cloud Computing ; Hadoop; Map Reduce; Scheduling.*

1. Introduction

Nowadays new technologies are evolved day by day in many fields like Internet of Things, Social Media, Organizations, e-commerce etc. This will create the large volume of digitized data; the generation of data rate is increased exponentially. The main problems have to deal with are the storage of this large volume of data they are stored in different systems. That means the data is distributed and can use the cluster of machines. Secondly, the type of the data that are generated may be structured, unstructured or semi-structured. Finally, the processing of these data is a big problem. The analysis of data is needed to take a decision from this large volume of data. The validity of data is to be maintained only the valid data is maintained [1][2].

A distributed system is a decentralized, reliable, scalable complex system, where the resources are scattered around different locations and are connected and coordinated their activities by the use of networks. This distributed system provides large capacity data storage. Cloud is like a distributed system environment that contains large capacity data centers in remote location. Cloud computing provides on-demand accessing of these shared resources via the internet. Some examples of service providers are Amazon, Microsoft, IBM, and Google Cloud are helped to do this activity easily and efficiently. The cloud resources are avail to the client, with the help of cloud service providers.

Cloud computing is a parallel and distributed environment contains a large number of interconnected virtual machines that are introduced into the environment dynamically. In order to make decisions, the distributed data has to be processed. Map Reduce is one of the programming models that are used for processing big data in data centers.

Scheduling is one of the main problems has to be dealt with, during the cloud resource sharing. It is the process of allocating resources to a particular job or allocation of virtual machines. Scheduling is an optimization problem.

This paper deals with the task scheduling in a cloud environment. It is considered as a two step procedure: mapping of virtual Machine (VM) and mapping from VM to physical resources.

This paper mainly focuses on scheduling of cloud resources. The rest of the paper is presented as follows. Section 2 gives introduction about MapReduce framework. Section 3 describes various scheduling algorithms for cloud. Section 4 deals with main Hadoop scheduling algorithms used in Job Trackers.

2. MapReduce Framework

Parallel processing and task scheduling are required for accessing cloud computing services. Traditional data processing applications are not suitable for processing Big data. MapReduce is one of the programming models that are used for processing big data in data centers. The data centers can include more than one map reduce jobs that are running simultaneously. It processes the data by dividing the job into independent tasks[3,[4].

The MapReduce contains two phases named Map and Reduce, and it produces a Key/Value pair as output. The MapReduce framework has a Master /Slave architecture. The master consists of Job Tracker and the slave contains TaskTracker. The JobTracker accepts jobs, divides it into task and distributes it to TaskTrackers. The TaskTrackers execute the task assigned by JobTrackers. The JobTrackers are in need of task scheduling. Scheduling is concerned with the optimal allocation of suitable resources for a particular task. The Map Reduce task scheduling is considered as an optimization problem [5]. Mainly the optimization of scheduling is based on the completion time of tasks that are to be scheduled.

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

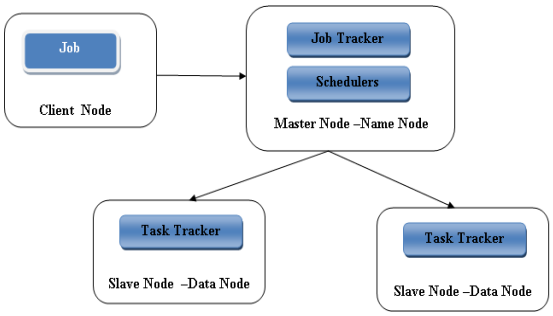


Fig 1: Basic Hadoop Framework

3. Task Scheduling Algorithms in Cloud Environment

The scheduling is considered as NP hard problem.

3.1. Earliest Deadline First (EDF) Scheduling

Virtualization is one of the main features of cloud computing. Virtual Machine Monitor (VMM) [6] known as Hypervisors maps the real time resources (physical resources) to Virtual Machine (VM). The virtual machine with earliest deadline has highest priority. The deadline calculation depends on Execution Time (E_t) and urgency factor (U_f) of VM using Eq. 1. Determine priority P (using Eq. (2)) in terms of the deadline of VM (D) and maximum deadline of VM in an entire schedule (D_{max}).The VM is having low execution time and high urgency factor got high priority and is scheduled. EDF provides high success rate. It results less profit to service providers and CPU utilization is not good.

$$D=E_t \times U_f \quad (1)$$

$$P=1 - (D/D_{max}) \quad (2)$$

3.2. Urgency Factor Prioritized (UFP) Scheduling

The VM priority assignment is based on deadline of job and the deadline is associated with only the urgency factor. The deadline D is calculated by summing up the deadlines of tasks in a particular job using Eq. 3. Determine priority P based on deadline alone using Eq. 4. UFP got a high success rate but give more profit to service providers. The CPU utilization is not efficient.[6]

$$\backslash D=\sum_{i=1}^{n} D_i \quad (3)$$

n- represents total number of tasks in VM .

$$P=1-U_f/U_{f\,max} \quad (4)$$

Where , U_f represents urgency factor of task that is going to execute and $U_{f\,max}$ Maximum urgency factor of schedule.

3.3. Multi-objective Particle Swarm Optimization (MOPSO) Scheduling

The muli-objective optimization (MOO) scheduling techniques [7] , MOPSO is a task scheduling algorithm in a cloud computing environment. It makes use of ranking strategy. The rank is to be calculated based on three objective values named Task Execution Cost (CE_t), Expected Completion time (C_t) and VM processing. It is calculated separately for each task. CE_t is the sum of the cost of executing tasks (PC) in a VM and the cost of transferring data files of task (IO_{Cost}).

$$CE_t=PC+IO_{Cost} \quad (5)$$

PC depends on processing time of the task and price of VM. IO_{Cost} is depending on the size of input and output files for the task and cost of bandwidth. C_t is computed by considering task length and VM processing speed. This algorithm reduces the waiting time and increase throughput and will lead to the improvement of execution time. The main shortcomings are the determination of processing speed of the VM and calculation of task size in terms of instructions are difficult.

3.4. Energy- Aware Task Scheduling (EATS)

EATS [9,10] algorithm mainly concentrates on the reduction of power consumption of utilizing resources and which in turn reduces the processing time. Consider the jobs with total computing load W_{total} can be divided as independent tasks with partition load of size in $chunk_i$ and they are not needed the synchronization and inter-task communication. EATS is a non-linear programming based model. The author shows that the resources that are idle and under utilizes consume more energy than fully utilized resources. The computing time for a worker node to perform $chunk_i$ units of load and with computing capacity μ_i is TP_i . The total execution time is TP.

$$TP_i = chunk_i / \mu_i \quad (6)$$

$$TP=\max TP_i \quad (7)$$

The total execution time is TP. EATS optimizes energy consumption and performance.

3.3. Naive Fair Sharing

The main aim is the fair sharing of cluster resources between jobs. These algorithms are applied independently to both map tasks and reduce tasks. Here the jobs are to be sorted in increasing order, according to the number of tasks in jobs. The tasks are to be allotted to a node when the data is available locally. [12]

3.4. Delay Scheduling

The delay scheduling makes use of ascending queue during scheduling. If the head of line job cannot launch a local task, skip it for a fixed amount of time and take the next job in the queue [12]. The main design concern is the determination of skip count D. The comparison is shown in following Table I.

TABLE 1: Cloud Scheduling Algorithms.

Name of Algorithm	Advantages	Disadvantages	Objectives	Approaches
EDF	High Success Rate	Less profit to service providers, CPU utilization is not good	Success Rate, CPU utilization ,Benefit for service providers	Urgency Factor Execution Time
UFP	More profit to service providers	Low Success Rate, CPU utilization is not good	Success Rate,, CPU utilization , Benefit for service providers	Urgency Factor
MOPSO	Reduces the waiting time , Increase through-	Determination of processing speed of VM is difficult, Calculation of task size in terms of	Execution time, Throughput	Ranking Strategy

	put ,Improve Execution Time	instruction is difficult		
EATS	Optimize Energy Consumption , Optimize Performance	Synchronization not obtained ,Inter-task communication not possible	Power consumption of Resources , Processing time	Non-Linear model
Naïve Fair Sharing Algorithm	Fair sharing of resources maintain data locality	Reduce task must read equal amount of data from all nodes., Head -of-line scheduling Sticky Slots	Fair sharing of resources	Fair sharing
Delay Scheduling	Increase Throughput, Nearly Optimal Data Locality	Applied to only sorted list of jobs, Locality associated with Skip Count	Data locality and fairness	Fair sharing

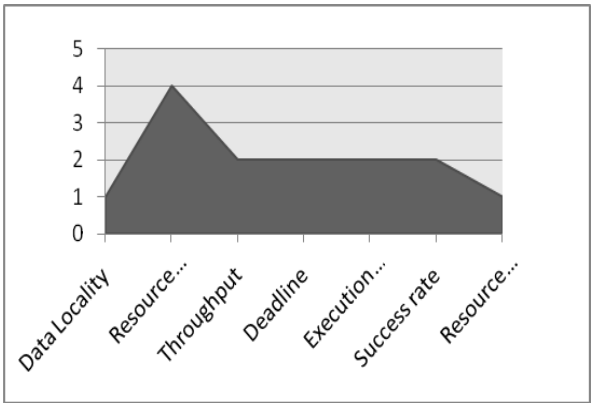


Fig 2: Performance Metrics Comparison Of Cloud Scheduling Algorithms

4. Hadoop Task Scheduling Algorithms

Hadoop is the Java based open source implementation of the MapReduce programming model and is used for implementing MapReduce. Hadoop scheduling algorithms are categorized as Inbuilt Scheduling Algorithms and User Defined Scheduling Algorithms. [12][13]

4.1. Inbuilt Scheduling Algorithms.

4.1.1. FIFO Scheduler

It is the default scheduling algorithm for Hadoop. The jobs are prioritized in first come, first served basis. The main drawbacks are the short job's response time is bigger compared to large jobs, low performance when running multiple jobs types; reduce data locality and starvation, not preemptive. The main advantages are simple and efficient, jobs are executed at the order they come.

4.1.2. Fair Scheduler

In fair scheduler all jobs get, on the average equal share of resources. It makes a group of job pools. It provides fairness in sharing resources in the pool. It allows quick response to small jobs among large jobs. It is complicated configuration and may lead to unbalanced performance because the job weights are not considered.

4.1.3. Capacity Scheduler

It ensures the fair management of the recourses among a large number of users. They are similar to fair schedulers, uses queues instead of resource pools. It maximizes the resource utilization and throughput. This is the most complex among three schedulers.

4.2. User Defined Scheduling Algorithms.

Many user defined scheduling algorithms are available. Some of them are as follows.

4.2.1. Delay Scheduling

In delay scheduling data is not available locally the task tracker will wait for a fixed amount of time [11][14]. The next task in the queue is scheduled if the above constraint is satisfied. It is mainly used in the homogeneous environment.

4.2.2. Longest Approximate Time to End (LATE)

Find out slow running tasks and create a speculative copy and complete the job. It improves the execution time [14],[15] ,[3]

4.2.3. Deadline Constraint Scheduler (DCS)

In DCS [3],[14],[16] the deadline is getting as part of the input. Jobs are scheduled if specified deadline is met. It is a dynamic scheduling scheme and can be used in both Homogeneous and Heterogeneous environments.

4.2.4. Resource Aware Scheduler (RAS)

RAS includes user free slot filtering and dynamic free slot advertisement [14]. It is a dynamic and it is for Homogeneous and Heterogeneous environments.

4.2.5. A Self-Adaptive MapReduce Scheduling Algorithm (SAMR)

SAMR [21] uses historical information recorded on each node and dynamically find slow tasks. It also considers the reaming exact time of task at the time of scheduling.

4.2.6. Tencent Distributed Data Warehouse Scheduling (TDWS)

The main objectives of TDWS [18] are the reduction of the starvation problem of small jobs, the urgent jobs are executed soon and long jobs finished in reasonable time. Here the jobs are arranged into two normal group name nrtgroup for large jobs and rtgroup for small jobs. There present another special group named keygroup which are reserved for urgent jobs. Nrtgroup is considered as the default group. The jobs in all these groups are handled in FIFO or fair sharing manner. Nrtgroup occupies more slots as compared to rtgroup. Keygroup has highest privilege, when a job comes in keygroup all the free slots are allocated to it and currently running jobs are preempted if needed. In order to attain data locality delay scheduling strategy is also incorporated with TDWS. The scheduling decision is also depends on the memory requirement of the job and its availability in datacenters. The main disadvantages include the calculation of the Skip Count value, availability of memory. It is also in need of the division of jobs.

4.2.7. Tolhit Scheduling Algorithm

Tolhit [19] find out the optimal node for scheduling speculative copy of slow tasks by considering historical information. The historical information about map and reduce weighs are maintained at each node. The parameter values associated with are mapped function execution weight (M1), reordering intermediate result weight (M2) and copy (R1), sort (R2) and reduce (R3) stage weights of reduce tasks. Based on this information the nodes are classified into k clusters using a genetic algorithm. The time to End (TTE) values of both map and reduce tasks are calculated by using this historical information. The stragglers are identified with TTE value and the speculative copy is assigned to node find out

by Tolhit algorithm. The assignment is also based on parameters like resource utilization the node and minimum spanning distances of the node from the scheduler. By the Tolhit overall delay is reduced and improves the execution time.

4.2.8. Constraint Programming Scheduler (CPS)

CPS is a constraint programming model and it consists of a CP scheduler; translate the jobs and available resource into an Optimization Programming Language (OPL). It is also in need of benchmarking tools and is defined according to the value of execution, index of CPU slots (based on CPU speed) and execution time of memory (how much memory is reserved for CPU slot) [18].

4.2.9. Multi-Objective Earliest Finish Time (MOEFT) Scheduling

In multi-objective EFTS [12], map tasks scheduled first followed by reduce tasks. These tasks may be scheduled on different VMs, according to the availably of slots. The tasks are to be selected as from queue in order. This is an iterative process. The map task without a parent is placed on the top of the queue. The earliest finish time depends on the number of tasks in job, scheduling policy and decision model. The workload information s got updated as a result of completion of the map and reduce task. The scheduling decision is based on the completion time with budget constraint and cost with deadline constraint. The execution time task t_i with budget b is T (t_i b) as in Eq. 8. It is an average value of the budget [20][21]

$$T(t_i\ b) = Total\ budget\ B / Total\ number\ of\ tasks\ T \quad (8)$$

The completion time of task t_i with budget b is t_i (b).

$$t_i(b) = max\ t_{i\,\epsilon J}\ \{ T(t_i\ b)\} \quad (9)$$

Where J number of jobs . The minimum cost to complete all tasks of job is $C_{(Nm,Nr)}$

$$C_{(Nm,Nr)} = min\ t_{i\,\epsilon J} \sum_{ti\,\epsilon T} Ci(Di) \quad (10)$$

Where $C_i(D_i)$ represent completion time of task with deadline, N_m is the number of map tasks and N_r is the number of reduce tasks. This algorithm results minimization of execution time, which in turn reduces the marksman of schedule and maximize throughput. Table 2 shows the comparison of various Hadoop scheduling algorithms.

TABLE 2: Various Hadoop Scheduling Algorithms.

Name of Algorithm	Advantages	Disadvantages	Objectives	Approaches
Delay Scheduling	Simple. Achieve Data Locality .	Not effective for large jobs. Limited slot per node.	Reduce jobs misses deadline.	Delay Scheduling strategy.
LATE	Minimize job response time	Not ensure reliability Not consider map and reduce task separately.	Improve response time	Speculative Task Execution.
DCS	Increase System Utilization. Focus on Optimization.	The deadline should specified by user. Need nodes have uniform cost.	Optimized performance	Constraint based
RAS	Improve Performance. Better Resource Utilization.	Lack of Preemption. Needs additional capability to manage dynamic capability	Optimized performance	Resource utilization
SAMR	Use of historical information to tune map and reduce weights.	Does not consider job types	Improve execution time	Stage weight parameter
CPS	Reduce jobs misses deadline.	Greater execution time. Scheduling is time consuming.	Reduce jobs misses deadline.	Constraint Programming
Tolhit Scheduling	Overall delay is reduced. Improves the execution time.	Collect historical information.	Execution time.	Clustering Historical data collection.
TDWS	Deals starvation problem of small jobs. Urgent jobs considered. Long jobs finished in reasonable time.	Jobs are divided. Skip Count value calculation. availability of memory.	Efficient Schedule. Reduce jobs misses deadline.	Delay Scheduling strategy. Availability of memory.
MOEFT	Minimize Execution Time. Reduces the Marksman.	Calculation of budget is important. Deadline calculation should be correct.	Execution Time . Marksman.	Earliest Finish Time .

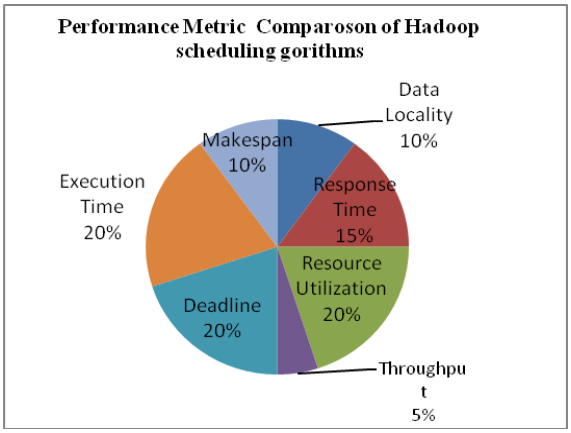


Fig 3: Performance Metrics Comparison of Hadoop Scheduling Algorithms

Performance metrics help to evaluate the effectiveness and performance of schedulers. Here Execution Time, Response Time, Data locality, Resource Utilization, Deadline, Make span, and Throughput are considered.

The various performance metric comparison of both basic cloud scheduling algorithms and Hadoop scheduling algorithms are shown in figure 2 and figure 3 respectively. From the comparison of basic cloud scheduling algorithms, it is concluded that they are mainly considered the proper and efficient resource utilization as the main performance objective. It results in the improvement of execution time, throughput, deadline requirement, etc. The main problem here has to face is the determination of the number of sharable resources over the internet. It is possible to improve

the resource utilization further by the prior knowledge of the amount of resources needed. This becomes a difficult process. This causes the inclusion of resource requirement prediction mechanism in association with scheduling strategy.

The comparison of various users defined Hadoop scheduling algorithms shown in figure 3, it is got that, they give more importance to performance objective like execution time, deadline, and resource utilization. This also indirectly results in the improvement of objectives like make span, response time, throughput etc.

5. Conclusion

Task scheduling is one of the important problems in cloud computing environment. Nowadays different scheduling algorithms are available for both cloud and Hadoop environment. They are having different performance objectives. They have their own advantages and disadvantages. The users and researchers use these algorithms directly or with some modifications according to their requirements.

References

[1] Ren Li , Haibo Hu , Heng Li "MapReduce Parallel Programming Model: A State-of-the-Art Survey" *International Journal of Parallel Programming,* Volume 44 Issue 4, August 2016 , Pages 832-866

[2] Roger Johannessen, Anis Yazidi, Boning Feng,"Hadoop MapReduce scheduling paradigms" *IEEE 2nd International Conference on Cloud Computing and Big Data Analysis* (ICCCBDA), 2017 ,pp. 175-179.

[3] Ehab Mohamed, ZhengHong, "Hadoop MapReduce job Scheduling Algorithms Survey ", *IEEE Conference Publications*, 2016 ,Pages: 237 – 242.

[4] Raja Manish Singh, Sanchita Paul, Abhishek Kumar ," Task Scheduling in Cloud Computing : Review ", IJCSIT) *International Journal of Computer Science and Information Technologies,* Vol. 5 , 2014, 7940-7944.

[5] Syed Arshad Ali, Mansaf Alam,"A Relative Study of Task Scheduling Algorithms in. Cloud Computing Environment", *2nd International Conference on Contemporary Computing and Informatics* (IC3I), 2016, Pages: 105 – 111

[6] Rekha Kashyap , Paritosh Louhan , Manish Mishra "Economy driven real-time scheduling for cloud", *10th International Conference on Intelligent Systems and Control (ISCO)*, 2016.

[7] Entisar S. Alkayal, Nicholas R. Jennings, Maysoon F. Abulkhair,"Efficient Task Scheduling Multi-Objective Particle Swarm Optimization in Cloud Computing ", *IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops)*, 2016, Pages: 17–24

[8] Suraj Pandey ,Linlin W, Siddeswara Mayura Guru, Rajkumar Buyya ,"A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments", *IEEE International Conference on Advanced Information Networking and Applications* ,2010,Pages 400-407.

[9] LeilaIsmail,AbbasFardoun, "EATS: Energy-Aware Tasks Scheduling in Cloud Computing Systems", *Procedia Computer Science,Elsevier,*Volume 83, 2016, Pages 870-877.

[10] Youwei Ding, Xiaolin Qin, Liang Liu, Taochun Wang,"Energy efficient scheduling of virtual machines in cloud with deadline constraint",*Future Generation Computer Systems , Elsevier*, Volume 50, September 2015, Pages 62-74

[11] M. Zaharia, D. Borthakur, J. S. Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling*," in Proc. EuroSys*, 2010, pp. 265–278.

[12] Nor Badrul Anuar,Arun Kumar Singaiah,Mohsen Marjani ," Multi-objective Scheduling of Map Reduce jobs in big data processing", *Springer , Multimed Tools Appl* may 2017

[13] Hadi Yazdanpanah,Amin Shouraki,Abbas Ali Abshirini,"A Comprehensive view of MapReduce Aware Scheduling Algorithms in Cloud Environments",*International Journal of Computer Applications,*Vol 127 No 6,October 2015, pp. 10-15.

[14] Mohd Usama, Mengchen, Liu , MinChen , "Job schedulers for Big data processing in Hadoop environment: Testing real-life schedulers using benchmark programs " *ELSEVIER Digital Communications and Networks* 26 August 2017.

[15] M. Zaharia, A. Konwinski, A. D. Joseph, R. H. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in *Proc. OSDI*, 2008, vol. 8, no. 4, pp. 29–42.

[16] Kamal Kc, Kemafor Anyanwu," Scheduling Hadoop Jobs to Meet Deadlines ", *IEEE Second International Conference on Cloud Computing Technology and Science* , 2010,Pages 388-392.

[17] Norman Lim , Shikharesh Majumdar , Peter Ashwood-Smith, "A Constraint Programming Based Hadoop Scheduler for Handling MapReduce Jobs with Deadlines on Clouds" *6th ACM/SPEC International Conference on Performance Engineering* 2015,Pages 111-122.

[18] Yanrong Zhao , Weiping Wang , Dan Meng ,"TDWS: A Job Scheduling Algorithm Based on MapReduce", *IEEE 7th International Conference on Networking, Architecture and Storage (NAS)*, 2012, Pages 313 - 317

[19] M. Brahmwar, M. Kumar, G. Sikka, " Tolhit – A Scheduling Algorithm for Hadoop Cluster ", *(ScienceDirect , ELSEVIER) Procedia Computer Science* ,Volume 89, 2016, Pages 203-208.

[20] M. Senthilkumar, P. Ilango , "A Survey on Job Scheduling in Big Data" ,*Cybernetics and Information Technologies, ACM,* Volume 16 Issue 3, 9 2016 ,pp. 35-51.

[21] Nagina, Dr. Sunita Dhingra ,"Scheduling Algorithms in Big Data: A Survey", *International Journal Of Engineering And Computer Science* ISSN:2319-7242 Volume 5 Issue 8 August 2016 Page No. 17737-17743

[22] Seokho Son and Kwang Mong Sim, "A Price- and-Time-Slot-Negotiation Mechanism for Cloud Service Reservations", *IEEE Transactions On Systems, Man, And Cybernetics—Part B: Cybernetics*, Vol. 42, No. 3, June 2012.

[23] Cong Wang, Kui Ren, "Towards Secure and Dependable Storage Services in Cloud Computing", *IEEE Transactions on Services Computing*, 1-7, 2011.

[24] Rafael Moreno-Vozmediano, "Multicloud Deployment of Computing Clusters for Loosely Coupled MTC Applications", *IEEE Transactions On Parallel And Distributed Systems*, Vol. 22, No. 6, June 2011.

[25] Sandeep Tayal "Tasks Scheduling optimization for the Cloud Computing Systems", *International Journal Of Advanced Engineering Sciences And Technologies*, Vol No. 5, Issue No. 2, 111 – 115

[26] Xiaoyong Tang, Kenli Li, Zeng Zeng, and Bharadwaj Veeravalli "A Novel Security-Driven Scheduling Algorithm for Precedence-Constrained Tasks in Heterogeneous Distributed Systems", *IEEE Transactions On Computers*, Vol. 60, No. 7, July 2011

[27] Xin Liu, Chunming Qiao, SUNY Buffalo Dantong Yu, Tao Jiang, "Specific Resource Provisioning for Wide-Area Distributed Computing", *IEEE Transactions On Computers*, Vol. 3, No. 7, July 2010