Study and Analysis of Various Task Scheduling Algorithms in the Cloud Computing Environment

Teena Mathew
Dept. of Computer Science & Engineering
Rajagiri School of Engineering & Technology
Kochi, India
helloteena@gmail.com

K. Chandra Sekaran
Dept. of Computer Science & Engineering
National Institute of Technology
Surathkal, India
kchnitk@ieee.org

John Jose
Dept. of Computer Science & Engineering
Rajagiri School of Engineering & Technology
Kochi, India
johnjose004@gmail.com

***Abstract*—Cloud computing is a novel perspective for large scale distributed computing and parallel processing. It provides computing as a utility service on a pay per use basis. The performance and efficiency of cloud computing services always depends upon the performance of the user tasks submitted to the cloud system. Scheduling of the user tasks plays significant role in improving performance of the cloud services. Task scheduling is one of the main types of scheduling performed. This paper presents a detailed study of various task scheduling methods existing for the cloud environment. A brief analysis of various scheduling parameters considered in these methods is also discussed in this paper.**

Keywords— Cloud computing; Task scheduling; Minimum execution time; Completion time; Cloudsim; Energy efficiency; Heuristics; Makespan.

I. INTRODUCTION

Cloud computing is a model for enabling omnipresent, commodious, on-demand access to a shared network containing a pool of configurable computing resources that can be easily purveyed and released with minimal management effort or service provider interaction [1]. Currently cloud computing is provides dynamic services like applications, data, memory, bandwidth and IT services over the internet. The reliability and performance of cloud services depends up on various factors like scheduling of tasks. Scheduling can be done at task level or resource level or workflow level. In this paper we are mainly focusing on the task scheduling approaches. Users send requests to the data center for computing jobs, named task. A task is a small piece of work that should be executed with in a given period of time. Task scheduling dispatches the tasks provided by the cloud users to the cloud provider on available resources.

Scheduling is performed on the basis of different parameters so that it increases the overall cloud performance. A task may include entering data, processing, accessing software, or storage functions. The data center classifies tasks according to the service-level agreement and requested services. Each task is then assigned to one of the available servers. In turn, the servers perform the requested task, and a response, or result, is transmitted back to the user.

Cloud task scheduling is a NP complete problem. In the process of task scheduling, the users submit their jobs to the cloud scheduler. The cloud scheduler inquires the cloud information service for getting the status of available resources and their properties and hence allocating the various tasks on different resources as per the task requirements. Cloud scheduler will assign multiple user tasks to multiple virtual machines. Good scheduling always assigns the virtual machines in an optimal way.

A good scheduling algorithm always improves the CPU utilization, turnaround time and cumulative throughput. Task scheduling can be performed based on different parameters in different ways. They can be statically allocated to various resources at compile time or can be dynamically allocated at runtime.

II. CLASSIFICATION OF TASK SCHEDULING

Based on the works relevant in literature [24], [2], [8], [11], [15], [17], [21], [25], [23]; we are classifying scheduling methods in cloud environment generally into three groups: resource scheduling, workflow scheduling and task scheduling, in which only the task scheduling approaches are focused in this paper. The entire classification is portrayed in Fig. 1. Resource scheduling performs mapping of virtual resources among physical machines and workflow scheduling is done to schedule workflows constituting an entire job in a suitable order. Task scheduling methods may be centralized or distributed. It can be performed in homogeneous or heterogeneous environment on dependent or independent tasks. In centralized scheduling a single scheduler is there to do all mappings whereas in distributed, scheduling is partitioned among different schedulers.

In case of distributed scheduling it has high implementation complexity. But here, processor cycles are saved; as the work load is distributed to partner nodes. Though centralized scheduling is easy to implement, it losses scalability and fault tolerance as it always has a bottleneck of single point of failure. Scheduling methods in distributed environment can be of two types: heuristic and hybrid techniques. Heuristic methods are classified into static as well

978-1-4799-3080-7/14/$31.00

as dynamic scheduling. Dynamic scheduling can be performed in online mode or batch mode. In static scheduling all the tasks are known a priori to scheduling and they are statically assigned to virtual resources. In dynamic scheduling all the tasks are scheduled instantly, as they arrive in the system. Dynamic scheduling mechanism performs better when compared to static algorithms. But the overhead of dynamic algorithms are high as we want to decide the schedule and update the system information instantly.

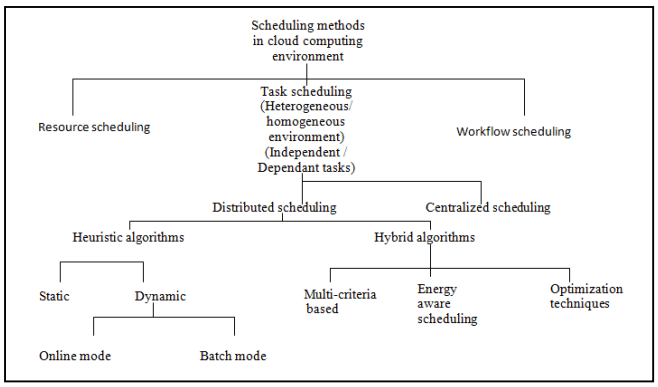

Fig. 1. Classification of Scheduling Methods in Cloud Computing Environment

The main intention of job scheduling is to achieve a high performance in computing and excellent throughput from the system. Static scheduling is easy to implement from programmer's aspect where as dynamic scheduling is more suitable for real world scenarios. Dynamic scheduling minimizes the cost needed to be paid for running the scheduler. Hybrid algorithms are three types: multi-objective, minimization-maximization approach or energy aware methods.

A. Scheduling Model in the Cloud Datacenters

Scheduling process in the cloud computing scenario has several fundamental components as shown in the Fig. 2.

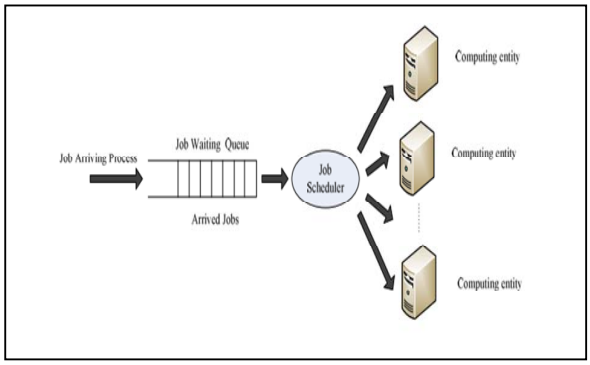

Fig. 2. Scheduling Model [22]

- Computing entity: This is provided by the implementation of virtualization technique in the cloud computing system. A number of virtual machines giving all the computing facilities like operating system, software etc. are present in the cloud system to process the submitted tasks. They are characterized by the computing capacity which indicates the number of instructions it can process in a second.
- Job scheduler: It is the important component of the scheduling process in the cloud computing environment. It determines the execution order of the jobs waiting in the queue.
- Job waiting queue: It is the queue in which the jobs are waiting to get assigned on a particular machine for execution.
- Job arriving process: It is the process in which jobs arrive into the scheduling system.

Rest of the paper is organized as follows. Section III presents the study of various existing scheduling algorithms in the cloud computing environment. It gives a brief analysis of some heuristic, energy efficient and hybrid task scheduling approaches. Section IV consolidates various scheduling methods and the parameters supported by them. The paper ends with conclusion and discussion of future work.

III. Existing Task Scheduling Algorithms

In this paper we are classifying task scheduling into three different categories as aforementioned. They are heuristic, hybrid and energy efficient task scheduling algorithms. Each of this categorization is explained briefly in the coming sub sections with their sub classifications.

A. Heuristic Task Scheduling Approaches

Heuristics scheduling [2], [27], [28]; provides an optimal solution in which it uses the knowledge bases for taking the scheduling decisions. Heuristic approaches can be either static or dynamic. First we will look at the static scheduling algorithms.

1) Static Scheduling Methods: Static scheduling algorithms consider that all tasks arrive at the same instant of time and they are independent of the system resource's states and their availability. The static heuristics include the basic simple scheduling strategies like First Come First Serve and Round Robin methods. FCFS methods collects the tasks and queues them until resources are available and once they become available the tasks are assigned to them based on their arrival time. It is less complex in nature but does not consider any other criteria for scheduling the tasks to machines.

On the other hand RR method uses the same FIFO technique for doing the scheduling of the tasks but it allots a resource for each task for a particular time quantum [21], [8]. After that the task is pre-empted and queued until its next chance for execution. Opportunistic load balancing is another heuristic method of scheduling in which it tries to schedule the tasks to the next available machines based on their expected

completion time. It will result in poor makespan even though it tries to utilize the resources equally making all machines busy at the same time.

Minimum Execution Time and Minimum Completion Time [8], [2] are other two heuristic strategies in which MET assigns tasks on the machines based on which machine it takes less execution time. It selects the best machine for execution but do not consider the availability of resources at the time of scheduling so load imbalance will occur. Minimum Completion Time Algorithm selects machines for scheduling the tasks based on the expected minimum completion time of tasks among all the machines available. It considers the load of the machine also before scheduling the task on that machine. The task may not have minimum execution time on the same machine. Completion time of a task on a machine can be defined as the sum of the execution time of the task on that machine and the ready time of that particular machine. Execution time is the actual time needed for executing a task.

Min-Min and Max-Min [21], [2] are two other heuristic methods used for task scheduling. Min-min heuristic selects the smallest task first from all the available tasks and assigns it to a machine which gives the minimum completion time (fastest machine) for that task. It increases the total completion time of all the tasks and hence increases the makespan. But it does not consider load of the machines before scheduling as simply assigning smaller tasks on faster machines. Here the expected completion time and execution time for a task are considered to be almost same values or close values. The long tasks have to wait for completing the execution of smaller ones. But the method improves the system's overall throughput.

Max-Min is similar to min-min except that it selects the longest task(with maximum completion time) first to schedule on the best machine available based on the minimum completion time of that particular task on all available machines. Here the smaller tasks have to starve and load balancing is also not considered. Anyway it increases the makespan and system throughput than the min-min strategy since the longest task determines the makespan of all the available tasks in the system. Hence in max-min the longer tasks can be executed first in faster machines as well as smaller tasks can be executed in parallel on other possible machines which results in better makespan and balanced load than the previous method.

Genetic Algorithm and Simulated Annealing are two other general methods in heuristic approach which is used to perform near optimal scheduling. In Genetic Algorithm approach [12] we perform four different operations, evaluation, selection, cross over and mutation. The initial population represents the possible mappings of the given task list on the available machines. Each job is represented as a vector in which each position of that vector represents a task in the task list. The value in each position represents the machine to which the task is mapped. Each job represents a chromosome. Every chromosome has a fitness value indicating the overall execution time of all the tasks (makespan) which is formed from the mapping of tasks to resources constituting that chromosome and it is selected such that it reduces makespan. This method uses past results with present results to get better possible mappings and survival of the fittest takes place.

Simulated Annealing is an iterative method which can be represented similar to genetic algorithm in which it starts with a single solution (mapping) selected from a random distribution. The initial version of SA is evaluated to get a better version. After mutation the new makespan is analyzed. If it is lower (better) than the previous one then replace the old one with the new makespan. Simulated Annealing finds poorer solutions than Genetic Algorithm. The features of genetic algorithm and simulated annealing can be combined to get a better scheduling solution [20], [26].

2) Dynamic Scheduling Methods: In dynamic scheduling methods [8], [21] tasks are dynamic in nature. Here tasks arrive at different points of time and it is dependent on the system machine's state. Dynamic scheduling algorithms are classified into two categories: (1) online mode and (2) batch mode. In online mode tasks are assigned instantly once they arrive in the system like most-fit task scheduling algorithm where as in batch mode tasks are collected as a group and scheduled at predefined times. Min-min, max-min, round robin are some examples for batch mode. MCT, MET, OLB belongs to online mode, and works similar to static algorithms.

Switching algorithm is another algorithm in which it switches between MET and MCT as per the load of the system. K-Percent Best is another heuristic of same kind in which , a subset of k computationally higher ranking machines is first selected during the scheduling process. A good value of k shows that it always assigns a task to a machine from this list only. This method leads to a better makespan compared to MCT. It preserves machines which are more suitable for yet-to-arrive tasks.

In batch mode along with max-min, min-min methods, another heuristic is called sufferage heuristic in which the tasks are scheduled based on a sufferage value. It is calculated from the first and second earliest completion times of a task. The sufferage values are compared for different tasks and the task with higher sufferage is selected for scheduling on a same resource [8], [2].

B. Energy Efficient Task Scheduling Approaches

The power management of a data center depends up on various factors and task scheduling is a significant one among them. The various task scheduling algorithms that mainly focuses on the power consumption reduction, increasing energy efficiency, performance improvement and cost reduction are given below.

In [3]; three algorithms are given which mainly focuses on how to handle a request from the users in heterogeneous systems. The first one is a benefit driven algorithm in which the tasks are assigned on the best server machines based on a

benefit value calculated. This works for heterogeneous networks. For homogeneous systems here they are proposing two methods: power best fit algorithm in which they consider the machine with least power consumption increment as its choice for scheduling the task. And the other one is load balancing approach in which load balancing is done based on the power frequency ratio of each resource. Power frequency ratio indicates the computing capacity of the server.

In [4]; they mainly focus on energy efficient job scheduling considering the traffic load balancing in cloud datacenters. They look on the traffic requirements of the cloud applications. In turn it minimizes congestion and communication delays in the network. In [5]; scheduling of tasks are done by combining network awareness and energy efficiency. It satisfies QoS requirements and improves job performance. It reduces the number of computing servers and avoids hotspots. Network awareness is obtained by using feedback channels from the main network switches. This method has less computational and memory overhead.

In [6]; an optimized scheduling strategy is implemented to reduce power consumption along with satisfying task response time constraints during scheduling. It is a greedy approach which selects minimum number of most efficient servers for scheduling. The tasks are heterogeneous in nature so that they constitute different energy consumption levels and have various task response times. Optimal assignment is based on minimum energy consumption and minimum completion time of a task on a particular machine.

In [14]; they propose a green energy efficient method of scheduling using the DVFS technique. Using Dynamic Voltage Frequency Scaling method it reduces the power consumption of infrastructure. Minimizing number of computing servers and time reduces energy usage and can improve resource utilization. The servers are run at different combinations of frequencies and voltages. This method efficiently schedules the tasks to resources without compromising the performance of the system meeting the SLA requirements and saving energy.

C. Hybrid Scheduling Algorithms

Many of these algorithms are novel or are developed on the top of some existing methods incorporating more scheduling parameters to improve the performance. Some of the existing works under this category are given below:

In [10]; they schedule tasks based on their cost to different resources. The cost of services varies for different tasks based on their complexity. The algorithm considers resource cost and processing capability of resources. They group tasks based on the processing capacity and selects some best resources to schedule them in such a way to reduce cost. This algorithm reduces the makespan and the processing cost when compared to other scheduling method [13]. In [9]; task's priority is calculated for scheduling them on various resources. Based on the different attributes of the tasks, priorities are calculated for the tasks and they are sorted based on that. Then they are assigned on the machine which produces the best completion time. Hence this algorithm improves performance by having better completion time.

In [16]; the tasks are partitioned into various groups and they are replicated to local middleware of the system. It makes the system fault tolerant and load balancing improves response time and resource utilization. Lexi search method is employed here to schedule the tasks to various resources along with reducing the cost. The task is assigned based on a probabilistic measurement which is calculated based on the availability of the resource and execution time of the task. Load balancing reduces the overhead created at the scheduler in each resource.

In [7]; they develop an algorithm based on traditional min-min algorithm which includes scheduling based on load of the servers as well as considering the user priority. The users are classified into two categories as VIP and ordinary users. Load is balanced based on the maximum loaded resource and the makespan of the system. The method shows a good gain in user satisfaction, makespan and resource utilization ratio.

In [17]; they give a modified algorithm for weighted least connection algorithm called dual weighted least connection algorithm. In this method the weights (processing capacity) of the servers are calculated dynamically and load of the servers are calculated based on the characteristics of the tasks assigned on that server. The algorithm gives better load balancing and system efficiency compared to WLC method.

In [18]; an algorithm is proposed based on the divisible load theory which aims to reduce the overall processing time of the tasks. Homogeneous processors are used here for which the load fractions and processing time for each task are calculated. The divisible load is partitioned among different servers and hence it enables the fastest completion of the tasks within a short period of time. This method improves the cloud providers benefit as well as quality of service. This method results good in terms of performance, total cost, delay cost, efficiency when compared to other random methods.

In [19]; they propose an algorithm which is a modification done on the improved max-min algorithm [21]. It is based on the expected execution time in which it assigns a task with average execution time on the machine which gives minimum completion time. The largest task determines the makespan of the system and sometimes it may be too large then it will increases the makespan of the system and create load imbalance. Hence instead of choosing largest task they choose average largest task or nearest average largest task. This method produces more fair load balancing and makespan than improved max-min method.

IV. ANALYSIS OF EXISTING SCHEDULING ALGORITHMS

The previously mentioned task scheduling algorithms consider different metrics for performing the scheduling operation. In most of the methods task scheduling is performed based on one or two parameters. A good scheduling algorithm always satisfy the requirements of users providing them a good quality of service and at the same time must consider the benefits at the cloud service provider side.

It should always try to reduce the cost and power consumption along with providing better performance [11]. Load balancing and energy consumption are two main parameters that should be considered in a scheduling algorithm. It should also provide fairness and security to the users when providing services. Developing a better algorithm by considering the combination of some significant parameters together always result in a good scheduling algorithm which can be deployed in a cloud environment for providing better cloud services to the users which can be considered as future enhancement [17]. An analysis on above scheduling methods and the different scheduling parameters that are supported by them, their advantages and disadvantages are consolidated in the Table I; given below.

TABLE I. COMPARISON OF EXISTING SCHEDULING ALGORITHMS

Scheduling Method	Parameters Considered	Advantages	Disadvantages
First Come First Serve	Arrival time	Simple in implementation	Doesn't consider any other criteria for scheduling
Round Robin	Arrival time, Time quantum	Less complexity and load is balanced more fairely	Pre-emption is required
Opportunistic Load Balancing	Load balancing	Better resource utilization	Poor makespan
Minimum Execution Time Algorithm	Expected execution time	Selects the fastest machine for scheduling	Load imbalanced
Minimum Completion Time Algorithm	Expected completion time, Load balancing	Load balancing is considered	Optimization in selection of best resource is not there
Min-Min, Max-Min	Makespan, Expected completion time	Better makespan compared to other algorithms	Poor load balancing and QoS factors are not considered
Genetic Algorithm	Makespan, Efficiency, Performance, Optimization	Better performance and efficiency in terms of makespan	Complexity and long time consumption
Simulated Annealing	Makespan, Optimization	Finds more poorer solutions in large solution space, better makespan	QoS factors and heterogeneous environments can be considered
Switching Algorithm	Makespan, Load balancing, Performance	Schedules as per load of the system, better makespan	Cost and time consumption in switching as per load
K-percent Best	Makespan, Performance	Selects the best machine for scheduling	Resource is selected based on the completion time only
Sufferage heuristic	Minimum completion time, Reliability	Better makespan along with load balancing	Scheduling done is only based on a sufferage value
Benefit Driven, Power Best Fit, Load Balancing	Energy Consumption, Cost, Load balancing	Power consumption is reduced and cost is reduced even more number of servers used	Other QoS factors and completion time of tasks are less considered
Energy efficient method using DVFS	Energy Consumption, Makespan, Execution time	Energy saving as per load in the system producing better makespan	Cost and implementation complexity can be make better in future
DENS	Traffic load balancing, Congestion, Energy Consumption	Communication load is considered and job consolidation is done to save energy	Consider only data intensive applications with less computational needs
e-STAB	Energy efficiency, Network awareness, QoS, performance	Load balancing and energy efficiency is achieved based on traffic load, congestion and delay are avoided.	QoS factors can be considered for improvement in overall performance
Task Scheduling & Server Provisioning	Energy Consumption, Task response time, Deadline	Energy is reduced meeting the deadline of tasks	Makespan and cost are less considered here
Improved Cost Based Algorithm	Processing cost, Makespan	Resource cost and computation performance is considered before scheduling	Dynamic cloud environment and other QoS attributes are not considered
Priority based Job Scheduling Algorithm	Priority of tasks, Expected completion time	Priority is considered for scheduling. Designed based on multiple criteria decision making model	Makespan, consistency and complexity of the proposed method can be considered for improvement
Job Scheduling based on Horizontal Load Balancing	Fault tolerance, Load balancing, Response time, Resource utilization, Cost, Execution time	Probabilistic assignment based on cost. Highest probable resource and task are selected for assignment.	Algorithm never mentions how the total completion time of the tasks will remain
User Priority guided Min-Min	Priority , Makespan, Resource Utilization, Load balancing	Prioritized is given to users improving load balancing and without increasing total completion time.	Rescheduling of tasks to perform load balancing will increase the complexit y and time
WLC based Scheduling	Load balancing, Efficiency, Processing Speed	Dynamic task assignment strategy proposed, task heterogeneity is considered	Considering only load balancing feature

Cost Based Multi QoS Based DLT scheduling	Load balancing, Makespan, QoS, Performance, Cost	DLT based optimization model is designed for getting better overall performance	Machine failure, communication overheads and dynamic workloads are not considered
Enhanced Max-Min Algorithm	Makespan, Load balance, Average execution time	Improves makespan and load balancing when large difference occurres in the length of longest task and other tasks or speed of processors	Parameters considered are limited and only theoretical analysis is performed

The main scheduling parameters considered in the previously mentioned methods are listed below:

- Makespan : It is the total completion time of all tasks in a job queue. A good scheduling algorithm always tries to reduce the makespan.
- Deadline : It is defined as the period of time from submitting a task to the time by which it must be completed. A good scheduling algorithm always tries to keep the tasks executed with in the deadline constraint.
- Execution Time : This is the exact time taken to execute the given tasks. Minimize execution time is the ultimate aim of a good scheduling algorithm.
- Completion Time : Completion time is the time taken to complete the entire execution of a job. It includes the execution time and delay caused by the cloud system. Minimizing completion time of tasks is considered by many of the existing scheduling algorithms.
- Energy Consumption : Energy consumption in cloud data centers is a current issue that should be considered with more care these days. Many scheduling algorithms were developed for reducing power consumption and improving performance and hence making the cloud services green.
- Performance : Performance indicates the overall efficiency given by the scheduling algorithm in order to provide good services to the users as per their requirements. A good scheduling algorithm should consider the performance at the user end as well as the cloud service provider end.
- Quality of Service : Quality of service includes many user input constraints like meeting execution cost, deadline, performance, cost, makespan, etc. All are defined in SLAs which is a contract document defined between the cloud user and cloud service provider.
- Load balancing : It is the method of distribution of the entire load in a cloud network across different nodes and links so that at a time no nodes and links remain under loaded while some nodes or links are overloaded. Most of the scheduling algorithms try to keep the load balanced in a cloud network in order to increase the efficiency of the system.

From the different task scheduling algorithms mentioned in this paper, the overall percentage of each scheduling metric considered in different methods is consolidated in Fig. 3. This analysis is limited to the methods explained in this paper. It helps to identify the scheduling attributes which are considered most and which all is less significant in different algorithms so that better algorithms can be developed by including least considered metrics or combining them with other metrics in existing algorithms to get a good overall performance.

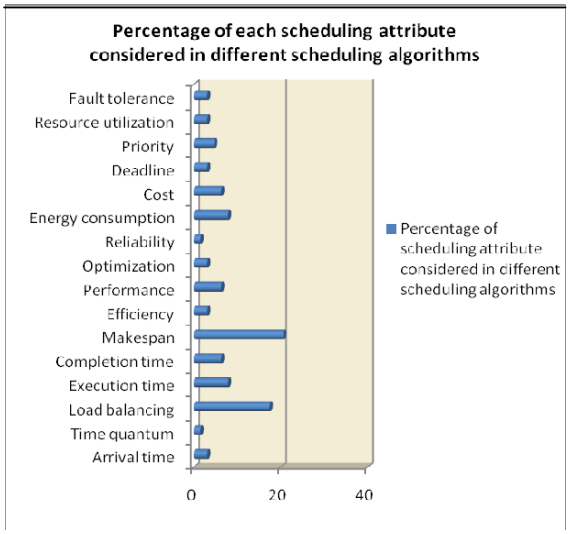

Fig. 3. Percentage of scheduling parameters considered in existing scheduling methods

From Fig. 3, we can see that makespan and load balancing are considered fairly in many methods for improvement. Completion time, execution time, performance and energy consumption are then considered in a moderate rate in the above analyzed algorithms. Some of the methods are scheduling based on priority, deadline, resource utilization, cost, efficiency etc. but only a few. Fault tolerance and reliability are also considered in fewer amounts in existing methods. Some algorithms give main focus to reducing the cost, and to have better resource utilization inside the cloud systems along with optimization. However the existing methods considered have failed to support above parameters together as it leads to high complexity and cost. Developing new scheduling algorithms considering more parameters together and hence producing better performance results can be considered as an important issue in current scenario. Existing methods and approaches can also be improved by incorporating more attributes thereby satisfying user SLA requirements and providing good quality of service.

V. Conclusion and Future Work

Efficient scheduling algorithms always play a significant role in the performance provided by a cloud computing

system. A study of existing task scheduling algorithms is done in this paper. It considers some heuristic, energy efficient and hybrid methods for study. A brief analysis of each method is done and most algorithms perform scheduling based on one or two parameters. A better scheduling algorithm can be developed from the existing methods by adding more number of metrics which can result in good performance and outputs that can be deployed in a cloud environment in future. The table created, consolidates all the different scheduling parameters used in the existing scheduling algorithms. A good scheduling algorithm must consider the requirements of users satisfying their needs provided in SLA and at the same time beneficial to the cloud providers. Combining different parameters such that to obtain an efficient scheduling algorithm and improve the overall performance of the cloud services can be done as an enhancement.

REFERENCES

[1] Peter Mell, Timothy Grance, "The NIST definition of Cloud Computing (September, 2011)", Accessed on May, 2014.

[2] Tracy D. Braun, Howard Jay Siegel, Noah Beck, "A Comparison of Eleven Static Heuristics or Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems", Journal of Parallel and Distributed Computing 61, pp. 810-837, 2001.

[3] Weicheng Huai, Zhuzhong Qian, Xin Li, Gangyi Luo, and Sanglu Lu, "Energy Aware Task Scheduling in Data Centers, Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications", volume: 4, number: 2, pp. 18-38, 2013.

[4] Dzmitry Kliazovich1, Sisay T. Arzo, Fabrizio Granelli, Pascal Bouvry and Samee Ullah Khan, "e-STAB: Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing", IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing e-STAB, 2013.

[5] Dzmitry Kliazovich, Pascal Bouvry, Samee Ullah Khan, "DENS: Data Center Energy-Efficient Network-Aware Scheduling, Cluster Computing", special issue on Green Networks, vol. 16, no. 1, pp. 65-75, 2013.

[6] Ning Liu, Ziqian Dong, Roberto Rojas-Cessa, "Task Scheduling and Server Provisioning for Energy-Efficient Cloud-Computing Data Centers", IEEE 33rd International Conference on Distributed Computing Systems Workshops Task, 2013.

[7] Huankai Chen, Frank Wang, Na Helian, Gbola Akanmu, "User-Priority Guided Min-Min Scheduling Algorithm For Load Balancing in Cloud Computing", IEEE, 2013.

[8] S.Nagadevi1, K.Satyapriya2, Dr.D.Malathy3, "A Survey On Economic Cloud Schedulers For Optimized Task Scheduling", International Journal of Advanced Engineering Technology, 2013.

[9] Xiaonian Wu, Mengqing Deng, Runlian Zhang, Bing Zeng, Shengyuan Zhou, "A Task Scheduling Algorithm based on QoS driven in Cloud Computing", Information Technology and Quantitative management, 2013.

[10] Mrs.S.Selvarani, Dr.G.Sudha Sadhasivam, "Improved Cost-Based Algorithm For Task Scheduling in Cloud Computing", IEEE, 2010.

[11] Isam Azawi Mohialdeen, "Comparative Study Of Scheduling Algorithms in Cloud Computing Environment", Journal of Computer Science, 9 (2): 252-263, 2013.

[12] Sung Ho Jang, Tae Young Kim, Jae Kwon Kim , Jong Sik Lee School, "The Study of Genetic Algorithm-based Task Scheduling for Cloud Computing", International Journal of Control and Automation Vol. 5, No. 4, December, 2012.

[13] Qi Cao, Zhi-BoWei, Wen-Mao Gong, "An Optimized Algorithm for Task Scheduling Based On Activity Based Costing in Cloud Computing", IEEE, 2009.

[14] Chia-Ming Wu, Ruay-Shiung Chang, Hsin-Yu Chan received, "A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters", Elsevier, 2013.

[15] Pinal Salot, A Survey of Various Scheduling Algorithm in Cloud Computing Environment, IJRET, February 2013, Available @ http://www.ijret.org/.

[16] Mousumi Paul, Debabrata Samanta, Goutam Sanyal, "Dynamic Job Scheduling in Cloud Computing Based on Horizontal Load Balancing", IJCTA, 2011.

[17] Lianying ZHOU, Xingping CUI, Shuyue WU, An Optimized Load-balancing Scheduling Method Based on the WLC Algorithm for Cloud Data Centers, Journal of Computational Information Systems, 2013.

[18] Monir Abdullaha, Mohamed Othmanb, Cost Based Multi QoS Job Scheduling using Divisible Load Theory in Cloud Computing, International Conference on Computational Science, ICCS 2013.

[19] Upendra Bhoi, Purvi N. Ramanuj, "Enhanced Max-min Task Scheduling Algorithm in Cloud Computing", International Journal of Application or Innovation in Engineering & Management, Volume 2, Issue 4, April 2013, pp 259-264.

[20] Shenai Sudhir, Survey on Scheduling Issues in Cloud Computing, Procedia Engineering, Elsevier, 2012.

[21] Archana mantri, Suman Nandi, Gaurav Kumar, Sandeep Kumar, High Performance Architecture and Grid Computing Computing, International Conference HPAGC 2011, Chandigarh, India, July 2011, Proceedings.

[22] Bo Yang, Xiaofei Xu, Feng Tan, Dong Ho, A Utility-Based Job Scheduling Algorithm for Cloud Computing Considering Reliability Factor, 2011 IEEE.

[23] Simsy Xavier, S.P.Jeno Lovesum, A Survey of Various Workflow Scheduling Algorithms in Cloud Environment, International Journal of

[24] Arya L.K, Verma A, Workflow scheduling algorithms in cloud environment - A survey, Recent Advances in Engineering and Computational Sciences, 2014, IEEE,1-4.

[25] Karan Dipakbhai Prajapati, Comparison of Virtual Machine Scheduling Algorithms in Cloud Computing, International Journal of Computer Applications12/2013;Volume83(973-93-80879-15-9):15.

[26] M. Coli, P. Palazzari, "Real Time Pipelined System Design through Simulated Annealing," Journal of Systems Architecture, vol.42, no. 6-7, 1996, pp. 465-475.

[27] J. Cao, D. P. Spooner, S. A. Jarvis, G. R. Nudd, "Grid Load Balancing Using Intelligent Agents", Future Generation Computer Systems, Vol. 21, No. 1, 2005, pp. 135-149.

[28] H. Izakian, A. Abraham, V. Snasel, "Comparison of Heuristics for Scheduling Independent Tasks on Heterogeneous Distributed Environments," in Proc. of the International Joint Conference on Computational Sciences and Optimization, IEEE, vol. 1, 2009, pp. 8-12.