An Algorithmic Improvisation in Task Scheduling for Cloud Environment.

Neema Mehulkumar Desai[1], Hardik J. Joshi [2], Dr. Chirag Suryakant Thaker [3]
[1]Dept. of Computer Science, Gujarat University, Ahmedabad, Gujarat, India
neemadesai02@gmail.com[1]
[2]Dept. of Computer Science, Gujarat University, Ahmedabad, Gujarat, India
hardikjjoshi@gmail.com[2]
[3]Information Technology Department, L.D. Engineering Collage, Ahmedabad, Gujarat, India
chiragthaker@yahoo.com[3]

Abstract: Supercomputers need vast computational resources. In Cloud Computing, to fulfill various intentions such as achieving the best performance by allocating appropriate task to the available resources, shortest response time, minimal total time for completion, optimal utilization of resources and many other forced the minds of the people to design, develop and propose a new effective and efficient scheduling algorithm. This algorithm is built based on RASA algorithm and the concept of Max-min strategy. This algorithm is developed to outperform scheduling process of RASA in case of total complete time for all submitted jobs. Proposed algorithm is based on expected execution time instead of complete time. So the scheduling tasks within cloud environment using this algorithm can achieve lower make span rather than original Max-min.

Keywords: Cloud Computing, Scheduling Algorithms, max-min algorithm, min-max algorithm, Resource Awareness Scheduling Algorithm (RASA).

I. INTRODUCTION

Cloud Computing can be defined as a new way of computing in which the scalable and virtualized resources are provided as a services over the Internet. Cloud Computing is actually an evolutionary approach that completely changed the way of delivering the services and giving best to the customers. "Cloud Computing consisting of a collection of inter-connected and virtualised computers that are dynamically provisioned and viewed as one or more unified computing resources based on service level agreement recognized through the negotiation between the service provider and consumers.[1]" The Figure 1 represents the various functionalities provided by the cloud computing environment such as its characteristics, challenges, benefits, service models, deployment models, cloud components and SLA characteristics. In addition, difference in computing sources in different nodes adds to the complexity of task scheduling. Furthermore, frequent data exchange among nodes, hosts, and clusters in data-intensive cloud applications makes the task-scheduling procedure extremely complicated.

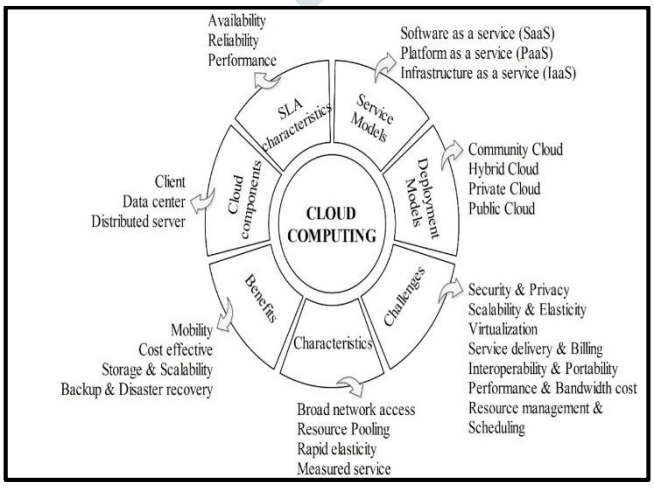

Figure 1. Fundamentals of Cloud Computing

Most of these methods focus on allocating CPU and memory resources to various cloud-computing tasks, assuming that all physical nodes and VMs have unlimited network bandwidth. This algorithm considers the limitation of resources, and provides resources according to task needs and resource loads. However, this algorithm did not consider the bandwidth requirements of tasks, nor did it consider the dynamic change of their resource requirements.

System resources like CPU, memory and bandwidth are used by many users, so it is little difficult to construct an efficient task scheduling algorithm. The efficiency of the algorithm is affected by many things like the processor power, speed, space and memory. Generally, task scheduling is the main process in infrastructure as a service model. While scheduling the task we consider virtual machines as scheduling machines. The main aim of task scheduling algorithms in cloud environment is to maintain the correct load on processors by considering the network bandwidth and increase their usage, efficiency and to reduce their task execution time.

II. TASK SCHEDULING IN CLOUD COMPUTING

As cloud consists of pool of resources and the users request the cloud to access its services. The advantage of job scheduling algorithm is to achieve a high performance computing and the best system throughput. The available resources should be utilized efficiently without affecting the service parameters of cloud. The requests of the users are needed to be scheduled in a proper manner so that the performance of the system can be increased. Scheduling involves various stages such as the **Resource Discovery Stage** (Discovering the resources that are available in the system), **Resource Selection Stage** (the selection of the resource is done based on some criteria) and **Task Submission Stage** (the task is submitted to the selected resource).

The selection of the resources can be done based on the parameters of scheduling which includes:

- **Makespan:** It is defined as the total completion time of all the tasks in a job queue. The makespan should be reduced to increase the performance of particular algorithm.
- **Fairness:** Determine whether user or applications are receiving a fair share of system resources.
- **Optimization:** Maximizing resource utilization as well as system throughput and minimizing the cost.
- **Deadline:** It is the period of time from submitting a task to the time by which it must be completed.
- **Load balancing:** It is the method of distributing the entire load in a cloud network across different nodes so that at a time no nodes remain under loaded. The load should be balanced to increase the efficiency of the system.
- **Scalability:** It is the ability of a computer application or product to continue to function well when it is changed in size or volume in order to meet a user need.
- **Response time:** The elapsed time between the end of an inquiry or demand on a computer system and the beginning of a response.

Based on these parameters the performance of the algorithm can be analysed.

Scheduling problem can be further classified into two categories depending on object O:

- **Optimization problem:** An optimization problem is based on finding the best solution among all the feasible solutions in set S.

- **Decision problem**: The aim of decision problem is that for a specified feasible solution $s \in S$, problem requires a positive or negative answer to decide whether the object O is achieved or not.

III. COMPARISON OF EXISTING SCHEDULING ALGORITHMS

In this paper we describing various task scheduling algorithms in a nutshell.

SCHEDULING METHOD	PARAMETERS CONSIDERED	ADVANTAGES	DIS-ADVANTAGES
First Come First Serve	Arrival time	Simple in implementation	Dosen't consider any other criteria for scheduling
Round Robin	Arrival time, Time quantum	Less complexity and load is balanced more Fairley	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 Considered	Optimization in selection of best resource is not there
Min-Min, Max-Max	Makespan, Expected, Completion time	Better Makespan Compared to other algorithms	Poor Load balancing and Qos Factors are not considered
Switching Algorithm	Makespan, Load Balancing, Performance	Schedulers as per load of the system, better makespan	Cost and time consumption in switching as per load
Task Scheduling & Server Provisioning	Energy, Consumption, Task response time, Deadline	Energy is reduced meeting the deadline of tasks	Makespan and coast 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 not selected for assignment	Algorithm never mention how the total completion time of the task 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 complexity and time
WLC based Scheduling	Load balancing, Efficiency, Processing speed	Load balancing, Efficiency, Processing speed	Dynamic task assignment strategy proposed, task heterogeneity is considered

Figure 2. Comparison of Existing Scheduling Algorithms

IV. PROPOSED SCHEDULING ALGORITHM

Our proposed scheduling algorithm, Improved RASA algorithm in Tsk Scheduling, is presented in Fig 3. The algorithm represents the completion time of the task *Ti* on the resource *Rj*. If the number of available resources is even, the Min-min strategy is applied to assign the first task, otherwise the Max-min strategy is applied. For instance, if the first task is assigned to a resource by the Min-min strategy, the next task will be assigned by the Max-min strategy. In the next round the task assignment begins with a strategy different from the last round. For instance if the first round begins with the Max-min strategy, the second round will begin with the Min-min strategy. Experimental results show that if the number of available resources is odd it is preferred to apply the Min-min strategy the first in the first round otherwise is better to apply the max-min strategy the first.

Alternative exchange of the Min-min and Max-min strategies results in consecutive execution of a small and a large task on different resources and hereby, the waiting time of the small tasks in Max-min algorithm and the waiting time of the large tasks in Min-min algorithm are ignored. As RASA is consist of the Max-Min and Min-Min algorithms and have no time consuming instruction, the time complexity of RASA is $O(mn^2)$ where m is the number of resources and n is the number of tasks (similar to Max-min and Min-min algorithms).

1. **for** all tasks T_i in meta task M_V
2. **for** all resources R_j
3. $C_{ij} = E_j + r_j$
4. **do** until all tasks in M_v are mapped
5. **if** the number of resources is odd then
6. for each tasks in M_v find the earliest complete time and the resources that obtains it
7. find the task T_k with the maximum earliest completion time
8. assign task T_k to the resources R_t that gives the earliest completion time
9. delete tasl T_k from M_v
10. update R_l
11. update C_{il} for all I
12. **else**
13. for each task in M_v find the earliest completion time and the resources that obtains it
14. find the task T_k with the minimum earliest completion time
15. assign task T_k to the resources R_l that gives the earliest completion time
16. delete task T_k from M_v
17. update r_l
18. update c_{il} for all i
19. **end if**
20. **end do**

Figure. 3 Proposed Scheduling Algorithm

Suppose that m resources $Rj(j = 1,...,m)$ have to process n tasks $Ti(i = 1,..., n)$. A schedule for each task is an allocation of one or more time intervals to one or more resources [16]. The expected execution time *Eij* of task *Ti* on resource *Rj* is defined as the amount of time taken by *Rj* to execute *Ti* given *Rj* has no load when *Ti* is assigned. The expected completion time *Cij* of task *Ti* on resource *Rj* is defined as the wall-clock time at which *Rj* completes *Ti* (after having finished any previously assigned tasks). Let *bi* denote to the beginning of the execution of task *Ti*. From the above definitions, $Cij=bi+Eij$. Let *Ci* be the completion time for task *Ti* and it is equal to *Cij* where resource *Rj* is assigned to execute task *Ti*. The makespan for the complete schedule is then defined as $Max_{Ti \in K} (C_i)$. Makespan is a measure of the throughput of the heterogeneous computing system (like computational grid)[9,11].

V. PROPOSED ARCHITECTURE

There are two options to run proposed algorithm, one is OpenStack and another is cloudSim. By using this both services we can easily access the all features and its advantage in our system and also environment too run our algorithm. CloudSim is framework of modeling and simulation of cloud computing infrastructure and services. It's been written in JAVA.

OpenStack is an open source Software Platform for cloudcomputing, mostly deployed as Infrastructure as a Service (IaaS), whereby virtual servers and other resources are made available to customers. The software platform consists of interrelated components that control diverse, multi-vendor hardware pools of processing, storage, and networking resources throughout a data-centre Users either manage it through a web-based dashboard, through command-line tools,or through RESTfull web services[12].

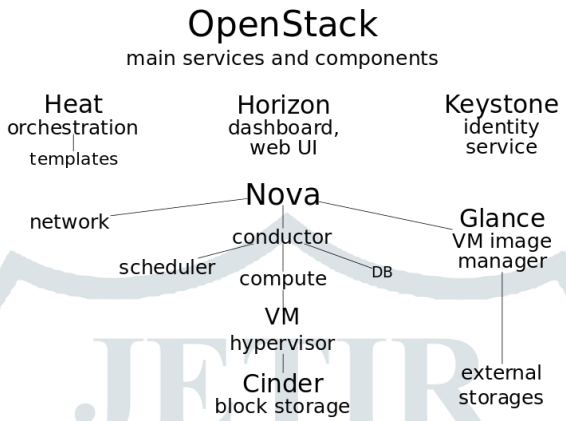

Figure 4. OepnStack Main Services.

CloudSim, is a framework developed by the GRIDS laboratory of University of Melbourne which enables seamless modelling, simulation and experimenting on designing Cloud computing infrastructures. CloudSim is a self-contained platform which can be used to model data centers, service brokers, scheduling and allocation policies of a large scaled Cloud platform. It provides a virtualization engine with extensive features for modelling the creation and life cycle management of virtual engines in a data center. CloudSim framework is built on top of GridSim framework also developed by the GRIDS laboratory [13].

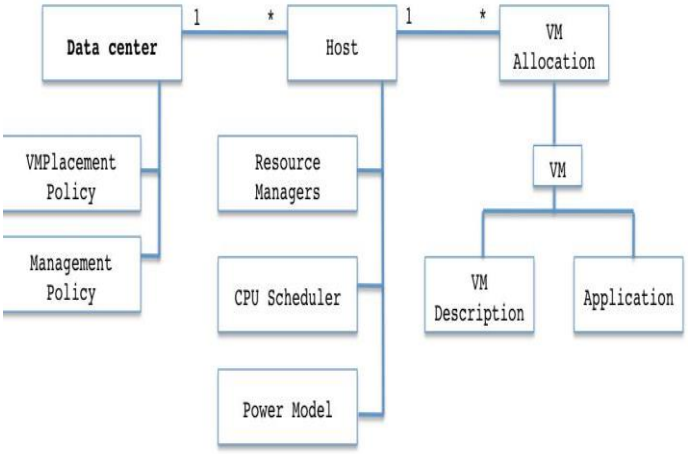

Figure 5. CloudSim Architecture

VI. OPEN ISSUES

Based on the survey on various task scheduling algorithms we came to know that there is still lots of improvements that can be carried out. The major issues in task scheduling algorithms are response time, cost, resource allocation, deadline, energy consumption and many other. Some techniques can be adopted to improvise the different issues and increase the performance of the system.

VII. CONCLUSION AND FUTURE SCOPE

Min-min and Max-min algorithms are applicable in small distributed systems. To achieve this, in this paper, a new task scheduling algorithm, Improved RASA algorithm in Tsk Scheduling, is proposed. Improved RASA algorithm in Tsk Scheduling is composed of two traditional scheduling algorithms; Max-min and Min-min. Improved RASA algorithm in Tsk Scheduling uses the advantages of Max-min and Min-min algorithms and covers their disadvantages. In this paper, the deadline of individual task, arriving rate of the different tasks, cost of the task execution on individual of the resource, cost of the communication and many other cases that can be a topic of research are not considered.

VIII. REFERENCES

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

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A.Rabkin, I. Stoica, and M. Zaharia. "A view of cloud computing", Communications of the ACM,53(4):50–58, 2010.

[3] Etminani. K, and Naghibzadeh. M, "A Min-min Max-min Selective Algorithm for Grid Task Scheduling". The Third IEEE/IFIP International Conference on Internet, Uzbekistan, 2007.

[4] Mrs.S.Selvarani1; Dr.G.Sudha Sadhasivam, improved cost-based algorithm for task scheduling in Cloud computing, IEEE 2010.

[5] Saeed Parsa and Reza Entezari-Maleki, "RASA: A New Task Scheduling Algorithm in Grid Environment" in World Applied Sciences Journal 7 (Special Issue of Computer & IT): 152-160, 2009.

[6] S. Devipriya, C. Ramesh, "Improved Max-Min Heuristic Model for Task Scheduling in Cloud". International Conference on Green Computing, Communication and Conservation of Energy (ICGCE), 2013

[7] El-Sayed T. El-kenawy, Ali Ibraheem El-Desoky, Mohamed F. Al-rahamawy "Extended Max-Min Scheduling Using Petri Net and Load Balancing" International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-4, September 2012

[8] Maheswaran, M., Sh. Ali, H. Jay Siegel, D. Hensgen and R.F. Freund, 1999. Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems. Journal of Parallel and Distributed Computing, 59: 107-131.

[9] Shu-Ching Wang, Kuo-Qin Yan, Shun-Sheng Wang, Ching-Wei Chen, "A Three-Phases Scheduling in a Hierarchical Cloud Computing Network", IEEE, 2011-888

[10] Braun, T.D., H. Jay Siegel, N. Beck, L.L. Boloni, M. Maheswaran, A.I. Reuther, J.P. Robertson, M.D. Theys and B. Yao, 2001. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, 61: 810-837.

[11] Wikipedia. Cloud computing. Retrieved from http://en. wikipedia.org/wiki/Cloud-computing,2011

[12] https://en.wikipedia.org/wiki/OpenStack

[13] http://www.cloudbus.org/cloudsim CloudSim