International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

An Optimized Task Scheduling Algorithm and Maintaining the Load in Cloud Environment

Nagashetty[1], Dr. Shiva Murthy G[2], Ramakrishna Prasad A.L[3]

[1]M. Tech Student, Department of CSE, VTU CPGS, Muddenahalli, Chikkaballapura, India

[2]Head of Department, Department of CSE, VTU CPGS, Muddenahalli, Chikkaballapura, India

[3]Assistant Professor, Department of CSE, VTU CPGS, Muddenahalli, Chikkaballapura, India

Abstract: ***Cloud provides convenient and on demand network access for computing resources available over internet. Individuals and organizations can access the software and hardware such as network, storage, server and applications which are located remotely easily with the help of Cloud Service. The tasks/jobs submitted to this cloud environment needs to be executed on time using the resources available so as to achieve proper resource utilization, efficiency and lesser makespan which in turn requires efficient task scheduling algorithm for proper task allocation. In this paper, we have introduced an Optimized Task Scheduling Algorithm which adapts the advantages of various other existing algorithms according to the situation while considering the distribution and scalability characteristics of cloud resources.***

Keywords: Cloud Computing, Makespan, Minimum/ Maximum Execution Time, Minimum/Maximum Completion Time, Load Balancing

1. Introduction

Now days a wide variety of computer ideas has been introduced i.e., Cloud supplies the user with sufficient and on-demanding contact of set-up for N no of computing resources available on the Internet. Cloud environment which has a widely scattered computing media that holds a huge content of virtualized computing sources that are near to the discrete or an institute. The actual motivation behind the implementation of CC is to supply a most promising worth able packages which is little bit stimulating. Scheduling of engagement in activity application and maintaining surfeit is just what was ordered a main am a source of in dwarf environment. This gave a pink slip be achieved by adopting proficient task scheduling algorithm. By over parameters a well- known as throughput, resource employment, charge, computational has a head start, pride of place, attitude, baud rate, resource availability and many preferably, time management algorithms are implemented.
By over parameters a well-known as throughput, resource employment, charge, computational has a head start, pride of place, attitude, baud rate, resource availability and many preferably, Scheduling algorithms are implemented. In decision to provide has a jump on Quality of Service (QoS), we prefer to bring about an capable hardship scheduling algorithm which maintains a valuable balance during resource hard pull to train efficiency, lesser derive span, concurrent task scheduling, having to do with resource hard pull and management. This complimentary is like the point of departure of latest course for scheduling task completely the at hand staple mutually analogy from the prompt one. Our intensify is to hast a portion of the tasks around the resources in an know ins and outs manner to move up in the world lesser derive span as compare with other urgent algorithms.

This paper is regarding the introduction of latest technique for scheduling task over the available resources with comparison from the existing one. My focus is to distribute the tasks over the resources in an appropriate manner to achieve lesser make span as compare with other existing algorithms.

Scheduling of tasks and allocation of resources are the important features of the cloud environment which directly affect the performance of a system. In order to achieve high throughput, various task scheduling algorithms have been introduced by the researchers for scheduling and scaling of resources. The adaption of the appropriate algorithm decides the performance of the system..

2. Methodology

- Job scheduling is done by adapting the existing cheduling algorithms. These algorithms when schedule tasks follow the same principal in each situation sometimes leading to increase in makespan of the processes.
- Time over all available resource is calculated for all processes; it tells about the total time taken by individual resource to execute all tasks.
- The task which needs to be migrated from one Resource to another so that makespan can be reduced.

A. Existing Algorithm

In cloud computing, all tasks have different features from each other. The main purpose of scheduling algorithm is fairly allocation of task over available resources and evenly distribution of workload as much as possible. To solve the problem of scheduling of tasks over available resources, numbers of algorithms have been introduced by researchers.

But many of those algorithms are not applicable in large scale distributed systems such as cloud environment or grid environment due to high communication cost. In this section, we will have a look over some existing. Scheduling algorithms which were introduced in order to serve qualitatively to the users. Some of them are:

i. Min-Min

It is a heuristic algorithm that starts by the whole of a apply of generally unmapped [14][1] tasks and employment by willingly finding the least possible expected anticipate of all tasks in meta-task. The hardship having the least possible expected closure time is occupied and situated the exact resource[1]. This hike is iterated until Meta-task is not empty. Here, carrying a lot of weight task has to warble for the end of the line of smaller ones.

ii. Max-Min

This is routinely used algorithm in abstracted environment which is gradually similar to the Min-min algorithm[14][1]. For starting this algorithm, expected cessation anticipates of each strain as using the accessible resource is calculated. A difficulty which has completely maximum closing time is scheduled far and wide a resource with completely minimum death warrants time. This race is regular until meta-task is not empty. Here, the waiting time of larger difficulty is reduced.

iii. RASA

RASA stands for "Resource Aware Scheduling Algorithm". RASA[15] is as a matter of course used algorithm in distributed environment which is gradually similar to the Min-min algorithm. For starting this algorithm, expected closing pioneer of each load as via the accessible resource is calculated. A responsibility which has completely maximum cessation time is scheduled round a resource with far and wide minimum capital punishment time. This stride is extended until meta-task is not empty. Here, the waiting time of larger difficulty is reduced.

iv. Improved Max-Min

The Max Min algorithm [11] schedules the duty as using the over maximum capital punishment time overall the resource that provides far and wide minimum cessation time. This algorithm supports fill balancing of accessible resources and allows concurrent death warrant of the submitted tasks. Total makespan is calculated everywhere larger responsibility execution.

v. Enhanced Max-Min

This algorithm instructed a modified Max-min algorithm [8]. This algorithm selects the difficulty mutually the respectable or nearest to sufficient execution anticipates instead of selecting largest tax for execution. This selected task is before scheduled during the resource by the whole of minimum closing time. This algorithm reduces the completely makespan and besides balances pall across resources.

B. Proposed Algorithm

a) Algorithm which adapts the advantages of various other existing algorithms according to the situation while considering the distribution and scalability characteristics of cloud resources.
b) Have introduced an Optimized Task Scheduling Algorithm which adapts the advantages of various other existing algorithms according to the situation while considering the distribution and scalability characteristics of cloud resources.
c) The proposed system uses
   1. Task scheduling in cloud computing.
   2. Makespan
d) Uses:
   Lesser makespan
   Maintaining Load Balance
   Less delivery time

3. Result

In the below system design Client gives N no. of file to the cloud operated by the Coordinator. Coordinator can receive the N no. of task as T1, T2, T3, T4, T5, ...Tn. Then the Coordinator can group the task based on the length of the task like group1 and group2 and next schedule the task is sorting each group by length as group1 and group2.After scheduling resource allocation is done by the Coordinator. The resources can be retrieved from the VM1 and VM2 the evaluation will be done in cloud by using constraints such as physical memory and CPU usage after evaluating these resource the maximum demand task group submitted to maximum capacity VM and the minimum demand task group submitted to minimum capacity VM respectively. Finally it will evaluate the VM efficiency these operation can be diagrammatically as shown in above figure1

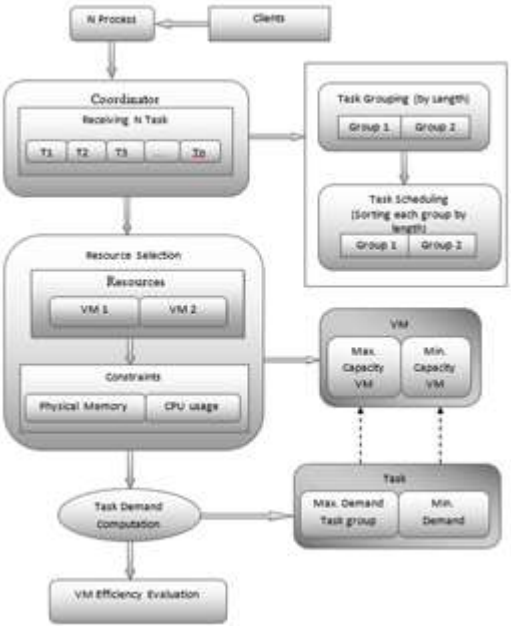

Figure 1: System Design

* ET = Execution Time
* TET = Total Execution Time
* MET = Minimum Execution Time
* SR = Slowest Resource
* FR = Fastest Resource

Figure 2: Optimized Task Scheduling Algorithm (Flowchart)

```
1.  for all submitted tasks Ti in meta-task Mv
2.  for all resources, Rj
3.  Compute Eij
4.  Yij = Σ Eij
5.  While meta-task is not empty
6.  for all tasks over fastest resource ri
7.  Zi = Yi – Ei
8.  for all task over slowest resource rj
9.  Aj = Ej>Zj
10. s1= min(Aj) // first optimized value
11. do if s1 >= Yi
12. s2 = min(Ej) // first MET over SR
13. s3 = second min(Ej)// second MET over SR
14. do if s2 >= Yi
15. Execute RASA
16. else if s3 >= Yi
17. Schedule min(Ej) over rj
18. Execute RASA for rest tasks over ri
19. else schedule min(Ej) over rj
20. else
21. s4 = Yj - s1
22. do if s1 >= s4
23. Schedule s1(Ei) over ri
24. else if s1 = max(Ej)
25. Schedule second max(Ej) over rj
26. else schedule s1(Ej) over rj
27. Execute Min-Min for rest task
```

* Ti: Tasks in meta-task Mv
* ri: Fastest Resource
* rj: Slowest Resource
* Eij: Execution Time of a task with respect to each resource
* Yii: Total Execution Time over Fastest Resource
* Zi: New calculated Execution Time over FR
* Ai: Execution Time over SR in different queue
* min(Ei), max(Ei): Minimum and Maximum ET over FR
* min(Ej), max(Ej): Minimum and Maximum ET over SR

Fig3. Optimized Task Scheduling Algorithm (Pseudo code)

4. Experimentation

Example and Analysis

Let us assume a meta-task *Mv*with five tasks T1, T2, T3,T4 and T5 and two resources R1 and R2. The processing speed and the bandwidth of communication links for each resource is shown in Table 1, whereas, Instruction Volume and Data Volume for each task is depicted in Table 2. The expected execution time of each task over all available resource is calculated. The results for the same are shown in Table 3.

Table 1: Resource Specification

Resource	Processing Speed (MIPS)	Bandwidth (MBPS)
R1	100	130
R2	800	60

Table 2: Meta Task Specification

Task	Instruction Volume (MI)	Data Volume (Mb)
T1	1400	95
T2	1600	66
T3	1200	47
T4	800	53
T5	1000	97

Table 3: Execution time of Tasks via Resource

Task	Resources	
T1	14	1.75
T2	16	2
T3	12	1.5
T4	8	1
T5	10	1.25

Time Evaluation

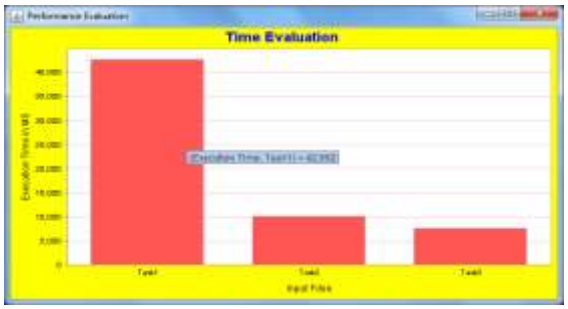

Figure 4: Time Evaluation

Memory Evaluation

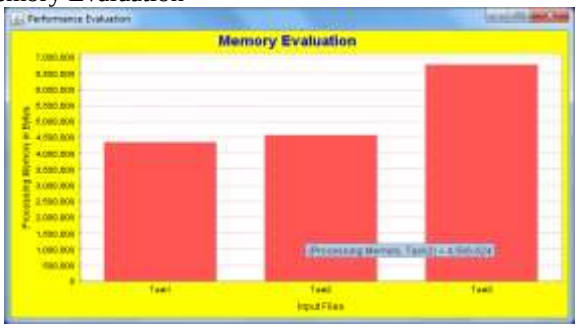

Figure 5: Memory Evaluation

Initial RAM Usage

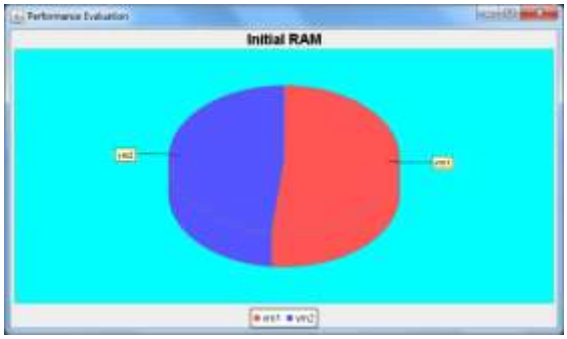

Figure 6: Initial RAM Usage

Initial CPU usage

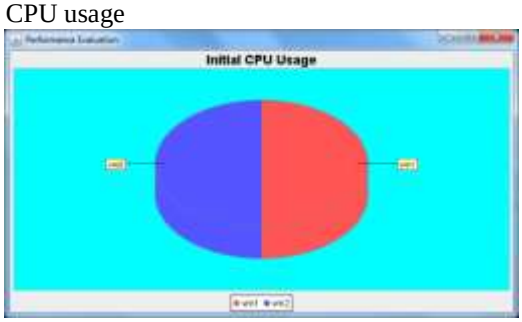

Figure 7: Initial CPU usage

RAM usage after allocation

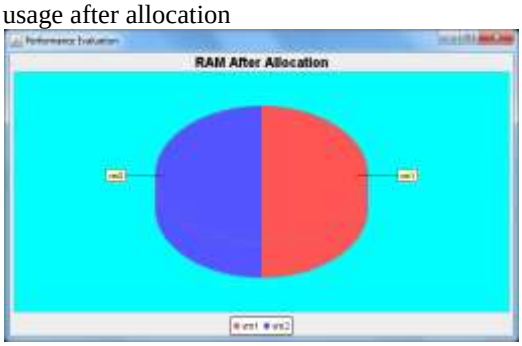

Figure 8: RAM usage after allocation

CPU usage after allocation

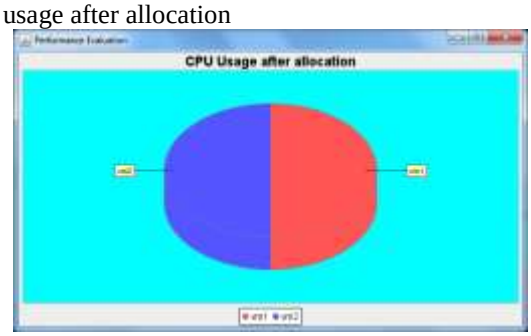

Figure 9: CPU usage after allocation

5. Conclusion

It is observed that the proposed algorithm improves resource utilization and completion time of tasks as compared to Sequential Assignment. The turnaround time of each job is minimized individually to minimize the average turnaround time. The results improve with the increase in task count. Processing power of virtual machines are computed using CPU and RAM properties. Hence, I achieved maximum utility of resources by allocating the maximum demand task group to maximum capacity virtual machine in cloud.

6. Acknowledgement

I would like to express my special thanks of gratitude to Assistant professor. Mr. Ramakrishnan Prasad A.L, Department of Computer Science and Engineering, Visvesvaraya Institute of Advanced Technology. who gave me the golden opportunity to do this wonderful project on the topic (An Optimized task Scheduling Algorithm and Maintaining Load in Cloud Computing), which also helped me in doing a lot of research and I came to know about so many new things I are really thankful to him. And, secondly I would also like to thank my parents who helped me a lot in finalizing this project within the limited time frame.

References

[1] BhavishaKanani and Bhumi Maniyar, “Review onMax-Min Task Scheduling Algorithm in Cloud Computing,” Journal of Emerging Technologies andInnovative Research, Volume 2, Issue 3, March 2015.
[2] Yash P. Dave, Avani S. Shelat, Dhara S. Patel andRutvij H. Jhaveri, “Various Job SchedulingAlgorithms in Cloud Computing: A Survey,” ICICES2014 - S.A.Engineering College, Chennai, TamilNadu, India, ISBN No. 978-1-4799-3834-6/14, IEEE2014.
[3] Maryam MasoudiKhorsand, Mehdi Effatparvar andMahdi Kanani, “A survey of Scheduling Algorithmsin Grid Computing,” International Journal ofResearch in Computer Applications and Robotics,ISSN 2320-7345, Volume 2, Issue 2, Pg. 118-126,February 2014.
[4] Rajwinder Kaur and PawanLuthra, "Load Balancingin Cloud System using Max-min and Min-minAlgorithm," International Journal of ComputerApplications (0975-8887), NCETCT-2014
[5] SunilkumarNakum, C. Ramakrishna, AmitLathigara, “Reliable RASA Scheduling Algorithmfor Grid Environment,” IEEE InternationalConference on Computer Communication andSystems (ICCCS), February 20-21, 2014.
[6] Sung-Min Jung, Nam-Uk Kim and Tai-MyoungChung, "Applying Scheduling Algorithms with QoSin the Cloud Computing," ISBN No. 978-1-4799-0604-8/13, IEEE 2013.
[7] S. Devipriya and C. Ramesh, "Improved Max-minHeuristic Model for Task Scheduling in Cloud," 2013
[8] UpendraBhoi, Purvi N. Ramanuj, “Enhanced Maxmin Task Scheduling Algorithm in CloudComputing,” International Journal of Application or Innovation in Engineering & Management, ISSN2319-4847, Volume 2, Issue 4, April 2013
[9] S Meraji, and R Salehnamadi, “A Batch ModeScheduling Algorithm for Grid Computing,” Journalof Basic and Applied Scientific Research, pp174-176,2013
[10] George Amalarethinam. D.I and VaaheedhaKfathee .S, "Max-min Average Algorithm for Scheduling Tasks in Grid Computing Systems," InternationalJournal of Computer Science and InformationTechnologies, Volume 3 (2), 2012,3659-3663

[11] O. M. Elzeki, M. Z. Reshad and M. A. Elsoud,"Improved Max-Min Algorithm in CloudComputing", International Journal of ComputerApplications (0975 – 8887), Volume 50-No. 12, July2012.
[12] Tarun Kumar Ghosh, RajmohanGoswami, SumitBera and SubhabrataBaran, "Load Balancing StaticGrid Scheduling Using Max-Min Heuristic," 2ndInternational Conference on Parallel, Distributedand Grid Computing , IEEE, 2012.
[13] T. Kokilavani and Dr. D.I.GeorgeAmalarethinam,"Load Balanced Min-Min Algorithm for static Meta-Task Scheduling in Grid Computing," InternationalJournal of Computer Applications (0975-8887),Volume 20-No. 2, April 2011.
[14] Kamalam .G.K and MuraliBhaskaran .V, "A NewHeuristic Approach: Min-mean Algorithm ForScheduling Meta-Tasks On HeterogeneousComputing System," International Journal of Computer Science and Network Security, VOL.10No.1, January 2010
[15] Saeed Parsa, Reza Entezari-Maleki, "RASA: A NewGrid Task Scheduling Algorithm," InternationalJournal of Digital Content Technology and itsApplications, vol. 3 (4), December 2009, pp. 91-99.