International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6, August, 2019

Load Balancing With Max-Min of Summations

Ranjan Kumar Mondal, Enakshmi Nandi, Payel Ray, Debabrata Sarddar

Abstract: ***Cloud computing is a computing tool for human-kind. In recent years, it is using to generate IT services, appliances for higher activities computing and outsourcing in a cost-efficient and flexible way. In modern times, a variety of types of bandwidth eater are growing speedily Cloud computing is growing phenomenal gradually to supply the different kinds of cloud services and applications to the internet-based customer. Cloud computing utilizes Internet applications to execute the large-scale jobs. The most important objective of cloud computing is to allocate and calculate different services transparently throughout a scalable network of machines. Load balancing is one of the significant issues in Cloud Computing. Loads should be divided as CPU load, the capacity of memory and system load which is the measurement of work that a computation system performs. Load balancing is a modern method where the load is being shared amongst several machines of a distributed system to enhance the utilization of various applications and response time of multiple tasks and prevent overloading situation and under loading situation. In or approach, we developed an algorithm, LBMMS, which combines all least completion time. For this study, LBMMS presents the proficient deployment of various resources in cloud computing.***

Keywords: Cloud Computing, Load Balancing, Distributed System, Scheduling.

I.INTRODUCTION

Load balancing with job Scheduling is considered to be an important issue in cloud computing [1]. The stipulate for resourceful load balancing with job scheduling helps to increase the computing performance. It is difficult to find an optimal solution related to job scheduling applications. The three main phases of load balancing are resource finding, collecting resource information and job completion [2].

Cloud computing is an example of a distributed application. Therefore clients put forward respective tasks to the cloud provider. Next Cloud provider informs the cloud information service broker about their various properties and applications. Cloud applications are registered in more than one cloud services. Cloud provider processes charges for scheduling various jobs according to multiple applications concerning job's necessity [3]. Cloud supplier ensures whether the task completed or not. Then it gathers the results and returns to the clients. Some load balancing algorithms exist to optimize longevity [4].

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Ranjan Kumar Mondal*, Computer Science & Engineering, University of Kalyani, WB, India
Enakshmi Nandi, Computer Science & Engineering, University of Kalyani, WB, India,
Payel Ray, Computer Science & Engineering, University of Kalyani, WB, India,
Dr. Debabrata Sarddar, Computer Science & Engineering, University of Kalyani, WB, India

All algorithms try to find different applications related to different jobs for minimizing the total execution time. This optimization method does not signify that it optimizes the authentic completion time of the definite job.

Two simple eminent algorithms utilized for load balancing are Min-Min [5] and Max-min [6]. These two algorithms effort for completion and execution time of distinct jobs related to available machines in cloud computing.

The Min-Min algorithm first determines the least completion time from several jobs [7]. Next, it desires the jobs with the optimum completion time among separate jobs. This method continues allocating the jobs with respective applications which generate the least execution time. The similar process is continuously replicated by Min-Min waiting for all tasks are scheduled.

The difficulties arise in the Min-Min algorithm is that it gives priorities to least application related jobs at first [8]. As an outcome, the scheduling process generated by the Min-Min algorithm is not satisfying until a number of minor tasks go beyond the large ones.

To reduce relevant complexity, Max-min algorithm schedules the bigger jobs first. But in several cases, the make-span may boost due to the completion of bigger jobs first [9]. The waiting time of minor jobs is enhanced in the Max-min algorithm.

Due to several weakness of the Min-Min algorithm, multiple well-improved scheduling processes have been discovered [10]. Different scheduling techniques have been discussed concerning makespan and proper load balancing process, to obtain an improved load balancing algorithm where the total response time of the system should be improved. our depicted algorithm consists of the strategy of Min-Min technique at the starting phase and then reschedules concerning highest completion time which is less than the make-span, obtained from the starting phase.

II.THE PROPOSED METHOD

To find out the matrix value as well as an arrangement of the job(s) vs. the machine(s) of an unbalanced matrix, we would consider a problem that makes a set of ‘m’ machines $M=\{M_{11}, M_{12},..., M_{1m}\}$. A group of ‘n’ jobs $J=\{J_{11}, J_{12},..., J_{1n}\}$ is considered to be allocated for completion on the ‘m’ existing machines and the performance value C_{ij}, where i=1, 2,...,m and j=1,2,..., n are mentioned within the value matrix wherever m>n. Initial of all, we tend to get the sum of every row and every column of the matrix, store the result in the array, named, Row-sum and Column-sum. Then we have to choose the first n rows with Row-sum, i.e., beginning with minimums to the next minimum to the array Row-sum and deleting rows equivalent to the remaining (r) jobs. Store the new array that ought to be the array for the primary sub-problem. Do again this method till remaining jobs less than a machine once the remaining jobs are but n, then, deleting (c) columns by Column-sum, i.e., equivalent to the value(s) most to next to make the last sub-problem.

Accumulate the ends up in the updated array that shall be the array for the last sub-problem.

III.PROPOSED WORK

Our proposed cloud scheduling algorithm name is LBMMS. The algorithm starts by executing the steps in total jobs first of respective every machine. It first distinguishes the job having the minimum completion time and the application producing it. Thus the job with the minimum completion time is scheduled first in LBMMS. After that, it assumes the minimum completion time from the time when some applications are scheduled with some jobs. While Min-Min opted the least jobs first it loads the fast executing application more which leaves the other applications idle. But it is easy for an excellent completion compared to further algorithms.

Algorithm:

Step 1*: First add all the jobs of each machine (column-wise) respectively.*

Step 2*: Then find a maximum total job.*

Step 2.1: *If we get two same maximum total jobs. Then select the machine that has a minimum unassigned job and assigns the selected job to that machine.*

Step 2.1.1: *If we get more than one minimum completion time of a particular machine then select the job of the corresponding summation of the total job is maximum.*

Step 3: *Next we have to select the unassigned machine that has minimum completion time value for the particular job selected in step 2 and then the job is dispatched to the selected machine.*

Step 4: *If we find more than one same value of the unassigned job again, then select that unassigned machine which corresponding machines have maximum value than another one. In the next step, this job is forwarded to a certain machine for execution.*

Step 5: *Do again Step 2 to Step 4, in anticipation of all jobs have been assigned completely.*

IV. CASE STUDIES:

There is a given example to solve the problem. Job T11, T12, T13, and T14 sub-jobs and N11, N12, N13, N14 machines are needed to be carried out for performing all jobs to assign different machines accordingly.

Table 1: gives the execution time for each job at different computing machines.

Job Machine	N_{11}	N_{12}	N_{13}	N_{14}
T_{11}	12	13	10	14
T_{12}	16	24	13	25
T_{13}	26	31	12	33
T_{14}	17	24	18	31

Step 1: First add all the jobs of each machine (column-wise) respectively.

Table 2: An illustration of four jobs needs to be processed is given.

Job Machine	N_{11}	N_{12}	N_{13}	N_{14}
T_{11}	12	13	10	14
T_{12}	16	24	13	25
T_{13}	26	31	12	33
T_{14}	17	24	18	31
Total Job	71	92	53	103

Step 2: Then find a maximum total job.

Table 3: The completion time of each machine is sorted priority basis as table 1.

Job Machine	N_{11}	N_{12}	N_{13}	N_{14}
T_{11}	12	13	10	**14**
T_{12}	16	24	13	25
T_{13}	26	31	12	33
T_{14}	17	24	18	31
Total Job	71	92	53	**103**

Step 3: Then add all sub-jobs of individual machines and find the maximum summation result of individual machines. Here we see that N14 is that machine which total sub-job result is high i.e. 103.

Table 4

Job \ Machine	N_{11}	N_{12}	N_{13}	N_{14}
T_{11}	12	13	10	**14**
T_{12}	16	24	13	25
T_{13}	26	31	12	33
T_{14}	17	**24**	18	31
Total Job	71	**92**	53	**103**

Step 4-5: Then select the minimum completion time of sub job of the machine N14 i.e. 14. Then Assigned this machine N14 with their minimum completion time 14. Then dispatch the respective job with corresponding machine N14 from the Meta job list. Next select the second higher sub-job summation value of rest of unassigned machines, i.e. 92 and then select the minimum sub-job completion value of that particular machine. The value is 13, but it will not consider because the corresponding value of row and column of value 14 of N14 should not be considered in this case due assignment of machine N14 has been done previously. Then go to value 24 but we get the same completion time value of machine N12. We consider first the next highest summation value that is 71, then see the next corresponding sub-job value in that column of both job value of T2 and T4. The corresponding value of T4 in the next higher summation unassigned machine N11 is 16 and the corresponding value of T4 in N11 is 17. So T4 is a higher value than T2. Thus select that 24 value of job T4 and assigned machineN12 with that job.Then select the next higher summation value i.e.71, then select minimum job completion time value is 12 but that row deleted previously and the next minimum completion time value is 16. Thus assigned job T2 of sub-job value 16 with machine N11.

Table 3: The completion time for each machine assigned accordingly

Job \ Machine	N_{11}	N_{12}	N_{13}	N_{14}
T_{11}	12	13	10	**14**
T_{12}	**16**	24	13	25
T_{13}	26	31	12	33
T_{14}	17	**24**	18	31
Total Job	**71**	**92**	53	**103**

Finally, select the last higher value that is 53. Here find the minimum completion time value is 10 that row already considers previously. Then choose the next minimum completion time value is 12 of job T3. Thus assign machine N14 of the sub-job value of T3 is 12.

Thus final completed the Meta –job list of different sub-job values with different assigned machines is entitled below.

Table 4: Final assigned machines with their respective completion time

Job \Machine	N_{11}	N_{12}	N_{13}	N_{14}
T_{11}	12	13	10	**14**
T_{12}	**16**	24	13	25
T_{13}	26	31	**12**	33
T_{14}	17	**24**	18	31
Total Job	**71**	**92**	**53**	**103**

Table 6: Optimal Result

V. COMPARISON

A completion time of each job at different computing machines displays following table.

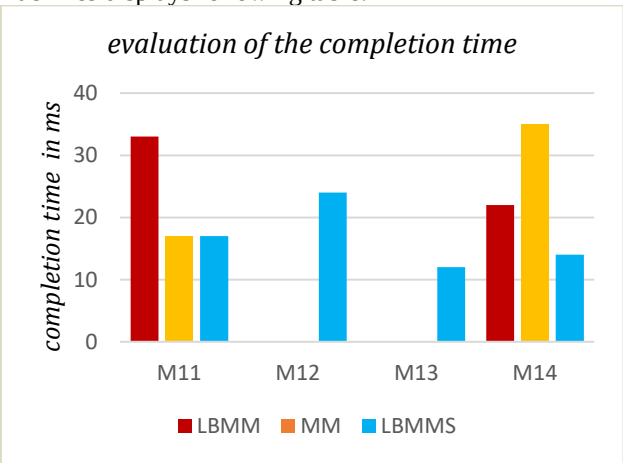

Fig 1: The Evaluation Of The Completion Time Of Each Job In The Singular Machine.

Figure 1 illustrates the completion time for each task at different computing nodes. To calculate the performance of our proposed approach is compared with another method by the case shown in Figure 1. It displays the comparison of the completion time of each computing nodes with our approach. The completion times for completing all tasks by using the proposed algorithm, LBMM and MM are 24, 33 and 35 ms, correspondingly. Our proposed approach display the minimum completion time and better load balancing than other existing algorithms.

VI. CONCLUSION

In the present paper, we presented a well-organized scheduling algorithm, LBMMS, for the networking to allocate all jobs to selected machines along with their resource ability. Our aim is to develop optimize scheduling technique for gaining minimum completion time and execution time and maintain load balancing properly. In the same way, our algorithm can obtain optimizing load balancing technique and give better act than further algorithms, such as MM, and LBMM from their analysis. In future studies, we want to apply our new technique in cloud computing application adversely.

REFERENCES:

1. Padhy, Ram Prasad. “Load balancing in cloud computing systems”, Diss. National Institute of Technology, Rourkela, 2011.
2. Amis, Alan D.,et. Al.. "Load-balancing clusters in wireless ad hoc networks." Application-Specific Systems and Software Engineering Technology, 2000. Proceedings. 3rd IEEE Symposium on. IEEE, 2000.
3. S. Wang, K. Yan, et.al. “Towards a Load Balancing in a Three-level Cloud Computing Network”, Proceedings of the 3rd IEEE ICCSIT, Chengdu, China, Sept. 2010, pages 108-113.
4. Hung, C.L. et.al. 2012, April. Efficient load balancing algorithm for the cloud computing network. In International Conference on Information Science and Technology, 2012, April (pp. 28-30).
5. Samal, Pooja, et. al. "Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing." International Journal of computer science and Information Technologies 4, no. 3 (2013): 416-419.
6. Mondal, Ranjan Kumar, et. al."Load balancing." Int. J. Res. Comput. Appl. Inf. Technol 4, no. 1 (2016): 01-21.
7. Mohanraj, Muthusamy. "International Journal of Emerging Technology and Advanced Engineering,” Reduction of Sags and Swells in a Distributed Generation System based on Environmental Characteristics." (2013): 382-385.
8. Kokilavani, T., and Dr DI George Amalarethinam. "Load balanced min-min algorithm for static meta-task scheduling in grid computing." International Journal of Computer Applications 20, no. 2 (2011): 43-49.
9. Mao, Yingchi, et.al. "Max–min job scheduling algorithm for load balance in cloud computing." In Proceedings of International Conference on Computer Science and Information Technology, pp. 457-465. Springer India, 2014.
10. Bhoi, Upendra, et.al. "Enhanced Max-min job scheduling algorithm in cloud computing." International Journal of Application or Innovation in Engineering and Management (IJAIEM) (2013): 259-264.

AUTHORS PROFILE:

Ranjan Kumar Mondal received his M.Tech in CSE from University of Kalyani, Kalyani, Nadia; and B.Tech in CSE from Government College of Engineering and Textile Technology, Berhampore, Murshidabad, WB under WBUT, WB, India. At present, he is a Ph.D. research scholar in CSE from University of Kalyani. His research interests include Cloud Computing, Wireless, and Mobile Communication Systems.

Enakshmi Nandi received her M.Tech in VLSI and Micro-electronics from Techno India, WB and B.Tech in ECE from JIS College of Engineering, WB under WBUT, West Bengal, India. At present, she is a Ph.D. Research scholar in CSE from University of Kalyani. Her research interests include Cloud Computing, Mobile Communication system, Device, and nanotechnology.

Payel Ray received her M.Tech in CSE from Jadavpur University, Jadavpur, and B.Tech in CSE from MCET, Berhampore, Murshidabad, WB under WBUT, WB, India. At present, she is a Ph.D. research scholar in CSE from the University of Kalyani. Her research interests include Cloud Computing, Wireless Adhoc and Sensor Network, and Mobile Communication Systems.

Dr. Debabrata Sarddar is an Assistant Professor at CSE from University of Kalyani, Kalyani, Nadia, WB, India. He completed his Ph.D. from JU. He did his M. Tech in CSE from DAVV, Indore in 2006, and his B.E in CSE from NIT, Durgapur in 2001. He has published more than 100 research papers in different journals and conferences. His research interests include Cloud Computing, Wireless, and Mobile Communication System.