Mapping of Independent Tasks in the Cloud Computing Environment

Biswajit Nayak[1], Sanjay Kumar Padhi[2]
Computer Sc. & Engineering
Biju Patnaik Technical University
Rourkela, Odisha, India

***Abstract*—Cloud computing is a technology that provides many resources and facility to share data. Due to the concept of open environment in the cloud computing the request or data increases quickly. So this problem can be solved by proper utilization of tasks along with available resources. Task scheduling algorithm plays an immense role in the cloud computing environment in minimizing the time required for completion of the task assigned to the resource available. There are several algorithms introduced to solve the problem of scheduling task of several kinds but all the developed algorithms are task dependent algorithms. The major criteria of the task scheduling algorithm are to optimize resource utilization in the diverse computing environment, so as to minimize makespan and execution time so that the accountability of healthcare industry that uses cloud computing can be enhanced. The proposed algorithm is designed to deal with variable length tasks by taking the advantages of the different heuristic algorithm and ensures optimum task scheduling with various available resources to enhance the quality of the healthcare system.**

Keywords—Scheduling; mixed model; cloud computing; makespan; healthcare

I. Introduction

Cloud computing environment differs from traditional computing environment on the basis of the target of scheduling. In the case of traditional computing, the transferred data is small but in the case of cloud computing, the transferred data is very large. Scheduling of resources ensures better service without any interrupt. This technique not only manages of task load but also fulfil the requirement of allocation of task dynamically with availability, flexibility, minimal cost and scalability features. The load balance technique ensures the availability of resource on demand, Proper utilization of resources under different conditions, reduced cost of resource use, Manipulation of energy at different load conditions [1][2].

Scheduling is a mechanism to maximize the throughput, required utilization of resource and also system performance through the allocation of task or job to the resource available. Due to increase in demand for technology with minimal cost and quick access time, some task scheduling prototype required. The processes start as the user submit tasks to the scheduler.

The scheduler schedules the task according to the availability of resources. As Fig. 1 shows in the job allocation process the scheduler allocates the job based on the cloud information repository. The datacenter computes the job within the stipulated time period. The task or job may be considered as data insertion, processing or accessing inserted data, software or it may be storage functions [3][4][5].

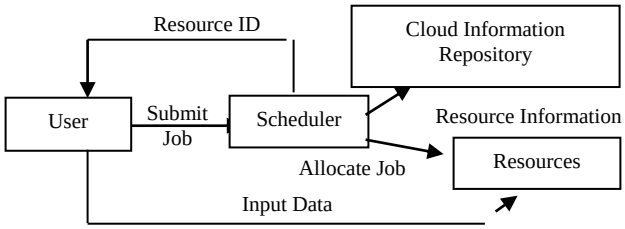


Fig. 1. Job Allocation Process.

II. Scheduling Parameters

There are certain cloud-computing performance metrics that are responsible for effective load balancing [6] [7][8]:

Throughput (TP): It is calculated to determine the performance of the system by calculating the number of tasks executed in one-unit time. It is measured by comparing with makespan. Increase in makespan reduces the performance and also decrease in makespan means optimum throughput.

Thrashing (TH): Thrashing takes place due to memory and other limited or exhausted resources. This occurs due to improper schedule of tasks, so good algorithm is required to maintain available resources.

Reliability (R): A system must be reliable to gain the faith of the user. If a task is transferred to any other virtual machine due to failure in task execution, then it can be treated as reliability of system. In other terms, a system is reliable if the system will work efficiently even if system fails to execute some of the task dedicated to.

Accuracy (A): This parameter ensures whether the result of the execution of the task meeting to the required result or not. If it will match the result, then it is accurate otherwise not.

Predictability (PR): The system should have the capability to predict the task allocation, execution and time required to complete considering available resources. It is also termed as a degree of prediction. It enhances the makespan of system.

Makespan (MS): It is defined as the time required completing all the tasks. In other words, it can be defined as the maximum time required by the system running over the data centre. Makespan is directly proportional to load balance;

means, if the makespan is less the load balancing, is good. One of the major characteristics of good task scheduling algorithm is diminishing makespan.

Scalability(S): It is a concept a system that ensures the execution of tasks under different conditional environment where the number of tasks may increase or decreases unexpectedly or periodically.

Fault Tolerance (FT): It is a method that increases the performance of a system by providing uninterrupted services even if one or more elements of the system fail to work properly. It is also responsible for resolving elements of logical errors.

Associated Overhead (AO): Associated overhead is directly proportional to load balancing. If the associated overhead is more that means the load balance is not proper. If the associated overhead is less that means the load of the system is proper.

Migration time (MT): It is a time required when a task is shifted from one resource to another or from one resource to different virtual machine or migration of service from one virtual machine to another virtual machine. The larger number of migration of the virtual machine leads to poor performance of system because it degrades the makespan.

Response time (RT): Time required acknowledging task for execution is known as response time. Lesser the response time leads to the greater performance of the system.

Energy Consumption (EC): It one of the major metrics in the cloud computing environment like makespan. Energy is calculated on the basis of energy consumed by all the devices connected to the system. The devices may be- output devices, device connectors, and application servers, etc.

Resource Utilization (RU): It is a concept defines the degree of use of resources in the system. If the load balancing is maximum that means resource utilization is maximized. Load balancing is directly proportional to resource utilization.

A good technique requires a good scheduler. Let's consider there are n numbers inputs for which N numbers of VMs are available. The set of tasks is (T1, T2, T3, T4... Tn). The heterogeneous environment in cloud computing uses expected time to compute matrix for load balancing. So the value of the matrix needs to be determined.

$$T_{EC_{ij}} = L_i / P_j$$

Where: L_i – Lenth of the ith task (MI)

P_j – The processing speed of the jth virtual machine (MIPS).

The most used performance parameter is the makespan in cloud computing. The different virtual machine takes different time for execution in the cloud computing environment. Good load balancing means minimal makespan.

The execution time of jth $VM(ET)_j$ is based on the decision variableX_{ij},

where

$$X_{ij} = \begin{cases} 1 \ If \ T_i \ \in \ VM_j \\ 2 \ If \ T_i \ \notin \ VM_j \end{cases}$$

The execution time of virtual machine literally depends upon the decision variable X_{ij}.

$$T_{E_j} = \sum_{i=1}^{n} X_{ij} * T_{EC_{ij}}$$

Makespan can be defined as the maximum time consumed by any virtual machine. So the makespan can be calculated as:

$$MS = Max[T_{E_j}]_{j=1}^{N}$$

III. Basic Task Scheduling Algorithms

Completion time is the basic criteria for MinMin algorithm. It uses two different time quantum-like Execution Time and completion time. Initially, it calculates the completion time, on the basis of minimum completion it finds the task and assigns to the corresponding resource. Then the task is removed and the completion time is updated. There is no longer waiting of processor for smaller tasks but it signifies the starvation for larger tasks and failed to perform well when small tasks are more as compare to large tasks. The Min-Min heuristics ensures the completion of task execution with minimum time period as compared to the other task and allocated to the suitable machine. According to the process the small tasks are assigned first then large tasks hence the makespan increase as the completion-time increases [9] [10].

Completion time is the basic criteria of the MaxMin algorithm. It uses two different time quantum-like Execution Time and completion time. Initially, it calculates the completion time, on the basis of minimum completion it finds the tasks and assigns to the corresponding resource on the basis of maximum completion time. There is no longer waiting of processor for larger tasks but it signifies the starvation for smaller tasks and failed to perform when the large task is more than a small task. So it eliminates the problem resides in MINMIN algorithm. Like the above algorithms, some other algorithms provide same result or poor with large search space. Some of the algorithms also tried to improve the makespan and throughput performance [11] [12] [13].

IV. Proposed Algorithm

The proposed scheduling algorithm tried to eradicate the problem persists in the basic algorithm developed. Even if the complexity of First-Come-First-Serve algorithm is very less, it performed less because it used arrival time to calculate time required to complete the task. Similarly, the Round Robin algorithm used arrival time along with the time quantum to reduce the time required to complete the task but still did not perform well. Apart from the basic algorithms some heuristics are used to enhance the performance. Some load balancing algorithm used to maximize the performance but due to the simultaneous use of resources or machines the performed poor makespan. Few more algorithms like Minimum Completion Time, Min-Min, Max-Min, suffrage algorithm etc. are used to solve the problem but still lacking to provide the optimum solution [14] [15] [16].

The proposed model is a Mixed Model to solve the starvation problem present in the model discusses above.

Step 1:

1. Start
2. Compute the completion time matrix of resources and tasks.

```
For all task Ti
    For all resources Rj
        CTij= ETij + rj
    End For
End For
```

Step 2:

3. Find the number of smallest number task "S" and largest number tasks "L"
4. If S>L

```
Go to Step 4
Else
Go to Step 3
```

Step 3:

5. For each task in the matrix, find the task ti with a minimum completion time and the resource on which it is calculated.
   Assign ti to resource Rj that has minimum completion time.
   Remove task ti from the matrix
   Update resource Rj ready time (rj)
   Update completion time of all un-mapped tasks in the matrix.
   Repeat all the steps of step3 until all the tasks in the matrix have been mapped.
   End For
6. Go to Step 5

Step 4:

7. For each task in the matrix, find the task ti with a minimum completion time and the resource on which it is calculated.
   Assign ti to resource Rj that has maximum completion time from selected minimum completion time.
   Remove task ti from the matrix
   Update resource Rj ready time (rj)
   Update completion time of all un-mapped tasks in the matrix.
   Repeat all the steps of step4 until all the tasks in the matrix have been mapped.
   End For
8. Go to Step 5

Step 5:

9. Display Result
10. Stop.

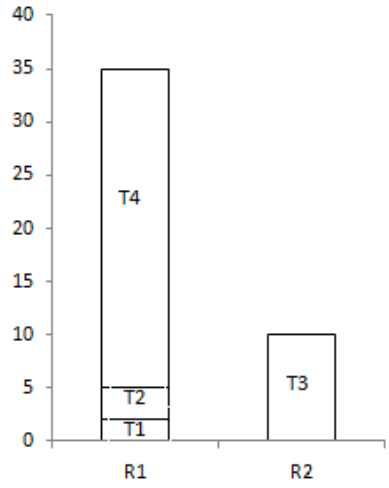
V. Performance Analysis

The proposed algorithm calculates the completion time of each task in a different machine and based on the expected completion time assign the tasks to the appropriate available resources. Let's consider four tasks (T1, T2, T3, T4) with execution time and two available resources (Table I). The table below clearly shows that the table consists of a large number of smaller tasks and a smaller number of large tasks.

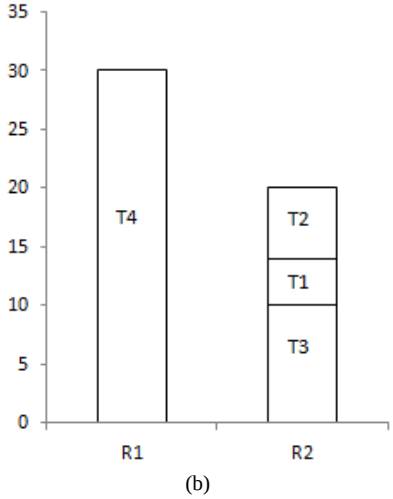
In Fig. 2(a) all tasks in are executed according to their minimum completion gives a makespan of 35 whereas the Fig. 2(b) executes or sort all the tasks according to their maximum completion time and give a makespan of 30.

TABLE. I. Resources for Allocation

Resource / Task	R1	R2
T1	2	4
T2	3	6
T3	4	10
T4	30	70



(a)



(b)

Fig. 2. (a) Output using STEP-3; (b) Output using STEP-4 NB: - X-axis showing the resources or machine (R1, R2) and Y-axis showing completion time (T1, T2, T3, T4).

Let's consider some inputs just opposite to the previous inputs and analyze the output to ensure that the algorithm performs better in all condition. Table II clearly shows that the table consists of a large number of large tasks and a smaller number of small tasks.

In Fig. 3, all tasks are executed according to their minimum completion gives a makespan of 121 whereas Fig.4 executes all the tasks according to their maximum completion time and give a makespan of 142. The example used consists of a large number of large tasks as compared to a number of small tasks. So it makes sure the execution of Step 3 of the algorithm and gives better makespan. Fig. 3 clearly shows the difference between the output of execution for Step 3 and Step 4. The makespan of Step3 is less as compare to the makespan of Step 4.

TABLE. II. RESOURCES FOR ALLOCATION

Resource / Task	R1	R2
T1	81	23
T2	112	32
T3	121	39
T4	61	17

On the basis of the above analysis for the better visibility Fig. 4(a) and Fig. 4(b) shows the output by comparing the time required for the execution of tasks at different conditions.

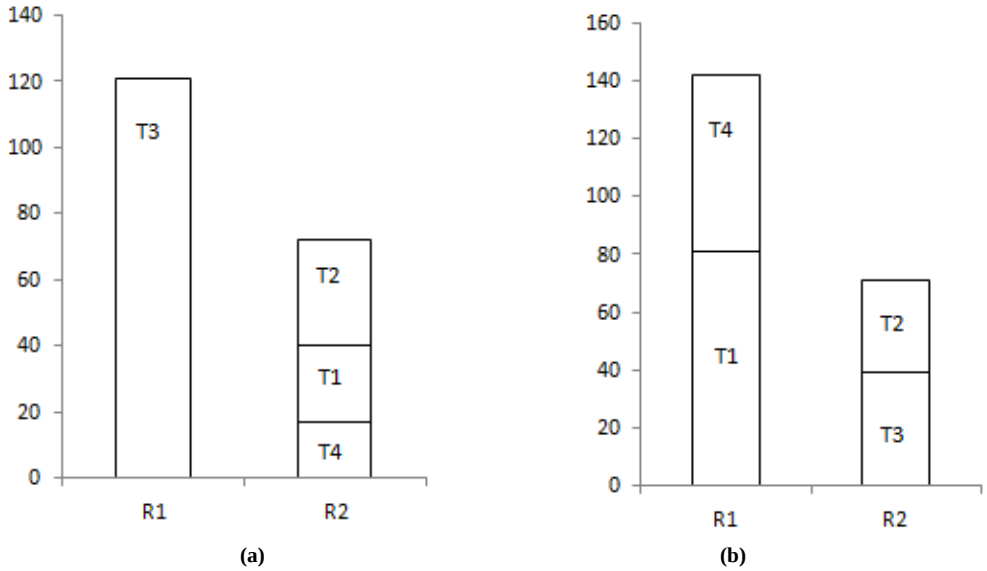
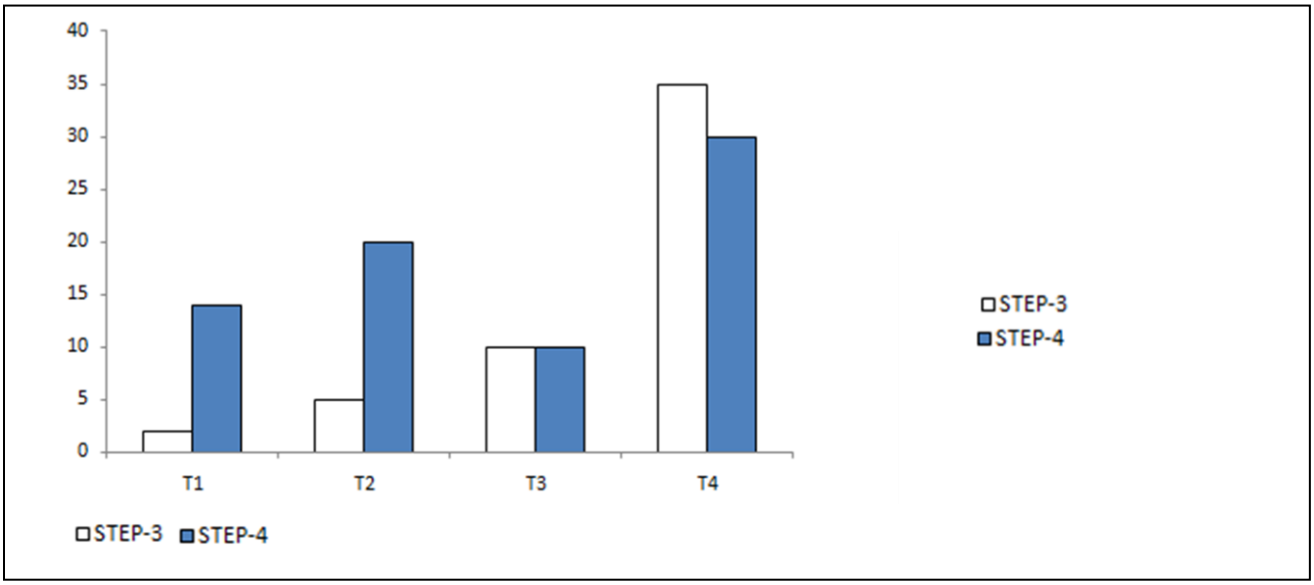


Fig. 3. (a) Output using STEP-3; (b) Output using STEP-4 NB: - X-Axis Showing the Resources or Machine (R1, R2) and Y-Axis Showing Completion Time (T1, T2, T3, T4).



(a) Execution Time Required at different Conditions NB: - X-Axis Showing the different Task Assigned and Y-Axis Showing Completion Time Required by different Tasks.

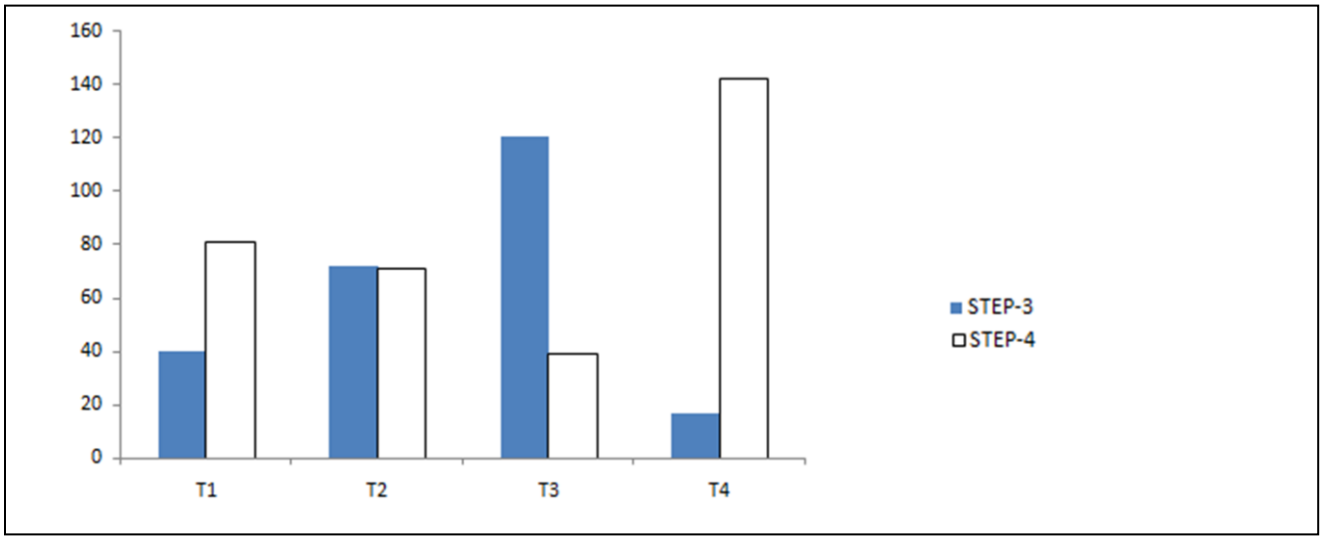


Fig. 4. Execution Time Required at different Conditions NB: - X-Axis Showing the different Task Assigned and Y-Axis Showing Completion Time Required by different Tasks.

VI. Conclusion

To realize the good performance of computing of scheduling of tasks in a cloud computing environment, a new algorithm is proposed. Different algorithms are tested for their suitability, feasibility, adaptability in the context of cloud scenario So that it can facilitate cloud-providers to provide a better quality of services. The proposed algorithm works on the problem exist when the number of small tasks is more in number or when the large tasks more in number. It performs in two phases. When the numbers of small tasks are more than the number of large size tasks then the algorithm will execute the large task first to increase efficiency to manage maximum completion time. In the reverse condition when the numbers of large size tasks are more than the number of small size tasks then the small tasks need to be executed first to increase the computing efficiency and to avoid starvation. In future the algorithm can be added with some other characteristics to enhance accountability.

References

[1] Tabak, E. K., Cambazoglu, B. B., & Aykanat, C. (2014). Improving the Performance of IndependentTask Assignment Heuristics MinMin, MaxMin and Sufferage. IEEE Transactions on Parallel and Distributed Systems, 25(5), 1244-1256. DOI:10.1109/tpds.2013.107.

[2] Nayak, B., Padhi, S. K., Pattnaik, P. K. (2017). Understanding the Mass Storage and Bringing Accountability. National Conference on Recent Trends in Soft Computing & It ´s Applications, pp. 28-35. ISSN: 2319 – 6734.

[3] Nayak, B., Padhi, S. K., & Pattnaik, P. K. (2018). Impact of Cloud Accountability on Clinical Architecture and Acceptance of Health Care System. 6th International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA). V.701, pp. 149-157. DOI 10.1007/978-981-10-7563-6_16.

[4] Suri, P. K. and Rani, R. (2017). Design of Task Scheduling Model for Cloud Applications in Multi-Cloud Environment. International Conference on Information, Communication and Computing Technology (ICICCT, Springer), 2017; 750:11–24.

[5] Mathew, T., Sekaran, K. C., & Jose, J. (2014). Study and analysis of various task scheduling algorithms in the cloud computing environment. 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI). DOI:10.1109/icacci.2014.6968517.

[6] Brinkerhoff, D. W. (2004). Accountability and health systems: Toward conceptual clarity and policy relevance. Health Policy and Planning, 19(6), 371-379. DOI:10.1093/heapol/czh052.

[7] Singh, P., & Kaur, N. (2016). A Review: Cloud Computing using Various Task Scheduling Algorithms. International Journal of Computer Applications, 142(7), 30-32. DOI:10.5120/ijca2016909931.

[8] Banga, P., & Rana, S. (2017). Heuristic-based Independent Task Scheduling Techniques in Cloud Computing: A Review. International Journal of Computer Applications,166(1), 27-32. DOI:10.5120/ijca2017913901.

[9] Singh, S., & Kalra, M. (2014). Scheduling of Independent Tasks in Cloud Computing Using Modified Genetic Algorithm. 2014 International Conference on Computational Intelligence and Communication Networks. DOI:10.1109/cicn.2014.128.

[10] Reda, N. M. (2015). An Improved Sufferage Meta-Task Scheduling Algorithm in Grid Computing Systems. International Journal of Advanced Research, 2015; 3(10): 123 -129.

[11] Kumari, E., & Monika, A. (2015). Review On Task Scheduling Algorithms In Cloud Computing. International Journal of Science, Environment and Technology, 2015; 4(2): 433 – 439.

[12] Jain, N. S.,(2016). Task Scheduling In Cloud Computing using Genetic Algorithm. International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR), 2016; 6(4): 9-22.

[13] Le, D., Bhateja, V., & Nguyen, G. N. (2017). A parallel max-min ant system algorithm for dynamic resource allocation to support QoS requirements. 2017 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON). DOI:10.1109/upcon.2017.8251134.

[14] Nayak, B., Padhi, S. K., & Pattnaik, P. K. (2018). Static Task Scheduling Heuristic Approach in Cloud Computing Environment. 5th Springer International Conference on Information System Design and Intelligent Applications. Vol. 862, pp.473-480. DOI: 10.1007/978-981-13-3329-3_44.

[15] Rajput, S. S., & Kushwah, V. S. (2016). A Genetic Based Improved Load Balanced Min-Min Task Scheduling Algorithm for Load Balancing in Cloud Computing. 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN). DOI:10.1109/cicn.2016.139.

[16] Bao, L. N., Le, D., Nguyen, G. N., Bhateja, V., & Satapathy, S. C. (2017). Optimizing feature selection in video-based recognition using Max-Min Ant System for the online video contextual advertisement user-oriented system. Journal of Computational Science,21, 361-370. DOI:10.1016/j.jocs.2016.10.016.