iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** VII **Month of publication:** July 2018

DOI: http://doi.org/10.22214/ijraset.2018.7131

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

An Optimized Hybrid Task Scheduling (OHTS) Algorithm based on Genetic Approach

Isha Sharma[1], Varun Jasuja[2], Dr. Rajesh Kumar Singh[3]
[1,2,3]*Computer Science, Guru Nanak Institute of Technology, Computer Science, SUS Institute of Computer Tangori*

Abstract: In this paper a unique modification to the existing TSCSA is proposed. In Improved algorithm largest job is selected and assigned to the resource which gives minimum completion time. *In order to achieve efficient QoS, a proper scheduling technique is required. Maintaining the resources is very important as scaling up or down of the resources occurs quite often due to the fluctuation of the resources used by the consumers. Therefore, a proper utilization of the resources becomes a key factor to obtain maximum QoS. Therefore, an efficient scheduling technique is required for utilizing the resources through parallelism such that the make span time and the cost can be improved. In this thesis work, an enhanced scheduling algorithm- Optimized Hybrid Task Scheduling (OHTS) algorithm is proposed which is based on the hybrid approach of that helps in allocating tasks to appropriate resources. Various algorithm always starts with task with minimum time and ignores the task with longer execution time. Hence an algorithm is needed which would overcome the shortcomings of these algorithms and provide better solutions in scheduling. The proposed algorithm helps in achieving better optimized results. The simulation results obtained shows that the performance of this proposed algorithm is more efficient as compared to Min-Min and TSCSA strategy in terms of the make span time and the energy.* ***The experimental results shows the new algorithms schedules jobs with lower make span.***

Keywords: cloud computing, Max-min algorithm, TSCSA, make span, minimum completion time, minimum execution time

I. INTRODUCTION

Cloud computing has become a new age technology that possesses large potentials in enterprises and markets. Cloud computing now's referred to as a supplier of dynamic services using terribly massive ascendable, on demand, virtualized resources over the net. It conjointly create it doable to access applications and associated information from anyplace. Industries are ready to rent resources from cloud for storage and alternative machine functions so their infrastructure value will be reduced considerably. They will conjointly create use of software's, platforms and infrastructures as a services, supported pay-as-you-go model.

Some of the drawbacks of the cloud computing is information Centre network structure expansibility, energy conservation, duplicate policies, security and planning mechanism [1] [2] [3]. The first objective of this paper is optimizing the planning policies. In cloud computing optimum planning is related to optimizing the resources being allotted. Due to the individuality of the model, resource allocation is performed with the target of minimizing the prices related to it. To create a collection of cloud services economical efficient supplier infrastructure one in all its needs is an efficient Task planning formula. Task planning formula is liable for mapping jobs submitted to cloud surroundings onto on the market resources in such the way that the entire latency, the make span is decreased [4] One of the feature of the Max-min algorithmic program is it selects the most important job and is dead on the quickest available resource. The most downside of the formula is it delays the execution of the smaller jobs and since of the dynamic nature of the cloud, execution of the smaller jobs could also be delayed indefinitely. Answer to the present is improved Max-min, it works well for the given set of jobs however dynamic cloud surroundings wherever the roles area unit submitted at any time the formula leads to degradation within the performance.

In this paper we have a tendency to be proposed two algorithms to extend the efficiency of improved Max-min. The remaining a part of the paper area unit organized as follows: section II presents some connected works on Max-min formula. Section III presents our planned changed 2 formulas on Improved Max-min algorithm, Section IV describes the Experimental results wherever we have a tendency to compare the results of the planned algorithm with alternative Max-min algorithms. Finally Section V concludes the paper and presents future work.

II. RELATED WORKS

A. Max-min Algorithm

This algorithm select the task based on task brust time. The task with maximum burst time is executed first and smaller tasks selected later.

B. Genetic Algorithm[5]

The genetic algorithms always provide the feasible solution to the problem. The genetic algorithm are defined with certain characteristic which are defined in a way to maximize the performance. Various GA algorithms are proposed by authors like CSA, honeybee, PSO, ACO etc.

C. Enhanced Max-min Algorithm [6]

Some times in a mata-task list task is simply too large compare to the opposite tasks. It should lead to increase within the overall make span. Here an answer is planned, first by choosing the work simply bigger than the common execution time and assignment to the slowest resource and so executing the improved Max-min algorithm. Max-min algorithmic program provides priority to the larger task, first it assigns the time intense jobs to the resources. The most disadvantage of this algorithmic program is early execution of the massive task would possibly increase the whole response time of the system. An answer to the current is that the improved Max-min algorithmic program that relies on the completion time. additional the result's improved in increased Max-min algorithmic program first executing largest job within the slowest resource and so using the improved Max-min. within the dynamic environment of the cloud, jobs are submitted batch wise. Always first designated largest job is that the largest job inside the batch, once there are multiple batches of job this algorithm doesn't offer the economical result.

II. PROPOSED ALGORITHM

Cloud computing is an emerging field due to its commercial importance and its role in information technology. One of the major issue is to utilize the cloud resources in a better manner to minimize the cost of cloud. Scheduling is one of the area to improves the cloud performance and reduce cost. Due to huge number of task instances, there is need of an algorithm that should focus on minimizing the make span time in order to enhance throughput. The algorithm should be capable enough to process large number of requests in given time frame.

The major need of task scheduling involves finding out a proper sequence to execute the tasks with available resources. We need to schedule tasks efficiently to reduce waiting time, response time, and execution time.

For major applications like OLAP, message passing and various multitier web application, modeling the order to execute tasks is very important to improve the performance.

There exist many algorithms for task scheduling like priority, Round-Robin, min-max and min-min, these algorithms claims to provide better solution to task scheduling. Many authors in exiting literature propose enhancement in in existing literature, but the scope of enhancement is still very high to utilize cloud in better mannered-scheduling is one of the most promising solution to these problems, many authors use re-scheduling to improve performance. Still the effective algorithms need to further improve cloud performance based on some hybrid approaches.

IV. METHODOLOGY

Initially the cloudsim is initialized, and after that all the entities are created (cloudlet, datacenter, broker).Then a datacenter is creator, which store list of created machines. Machines may contain more than one cure. Then a broker is created which schedules the tasks to Virtual Machine. The Virtual Machine (VM) is submitted to broker, and broker will schedule cloudlets to Virtual Machines. Using the values (cloudlets parameters and VM parameters) we calculate ECT (Expected Completion Time) for all cloudlets.

$$ECT=\text{Execution time} + \text{Waiting Time}$$

The computed ECT is used to find average completion time (ACT),

$$ACT= ECT/ (\text{Tasks} +VM)$$

Then, select any task cloudlet from cloud list, denoted by T_k. The T_k is cloudlet with the highest time. If T_k > ACT, then it is assigned to its corresponding machine for execution. Then T_k is deleted from the cloudlist and matrix is updated. The metalist is again checked for metatask and another task with highest burst time is selected and directly assigned to fast virtual machine. Again the Task is deleted and list is updated.

The proposed work is defined in two phases

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

A. Improved (proposed algorithm)

1) Algorithm

a) For all submitted tasks in Meta-task; Ti
b) For all resources; R
c) Cij = Eij + r
d) While Meta-task not Empty
e) 1 Find task Tk costs Arithmetic Average or nearest Greater than Arithmetic Average execution time.
f) 2 Assign task Tk to resource Rj which gives minimum completion time (Slowest resource)
g) Move task Tk from Meta-tasks set.
h) Update rj for selected Rj.
i) Update Cij for all j.

B. Improved Algorithm 2 on Max-Min Task Scheduling Algorithm (proposed)

1) Algorithm

a) For all submitted tasks in Meta-task; Ti
b) For all resources; Rj
c) Cij = Eij + rj
d) While Meta-task not Empty
e) Find task Tk costs Geometric Average or nearest Greater than Geometric Average execution time. Assign task Tk to resource Rj which gives minimum completion time (Slowest resource).
f) Remove task Tk from Meta-tasks set.
g) Update rj for selected Rj.
h) Update Cij for all j.

V. EXPERIMENTAL RESULTS

The proposed scheme is quite enhance in terms of energy and.make span time. The efficient task scheduling results in enhancing the existing performance. The results are shown in Figure 1 given below:

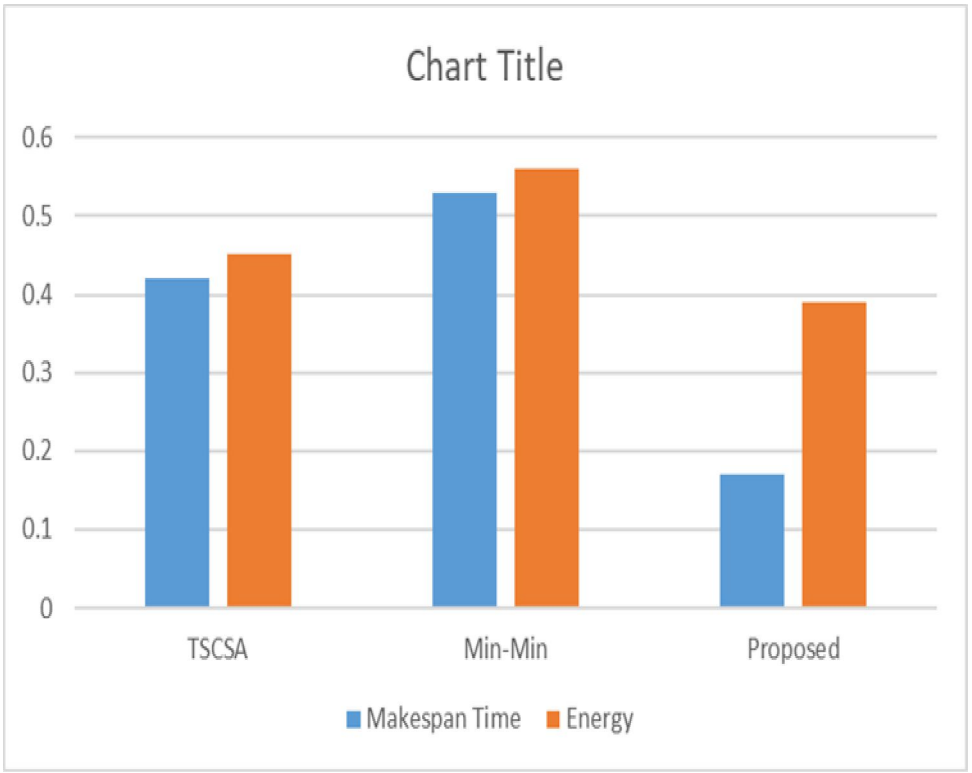


Figure 1 Makespan Time and Energy

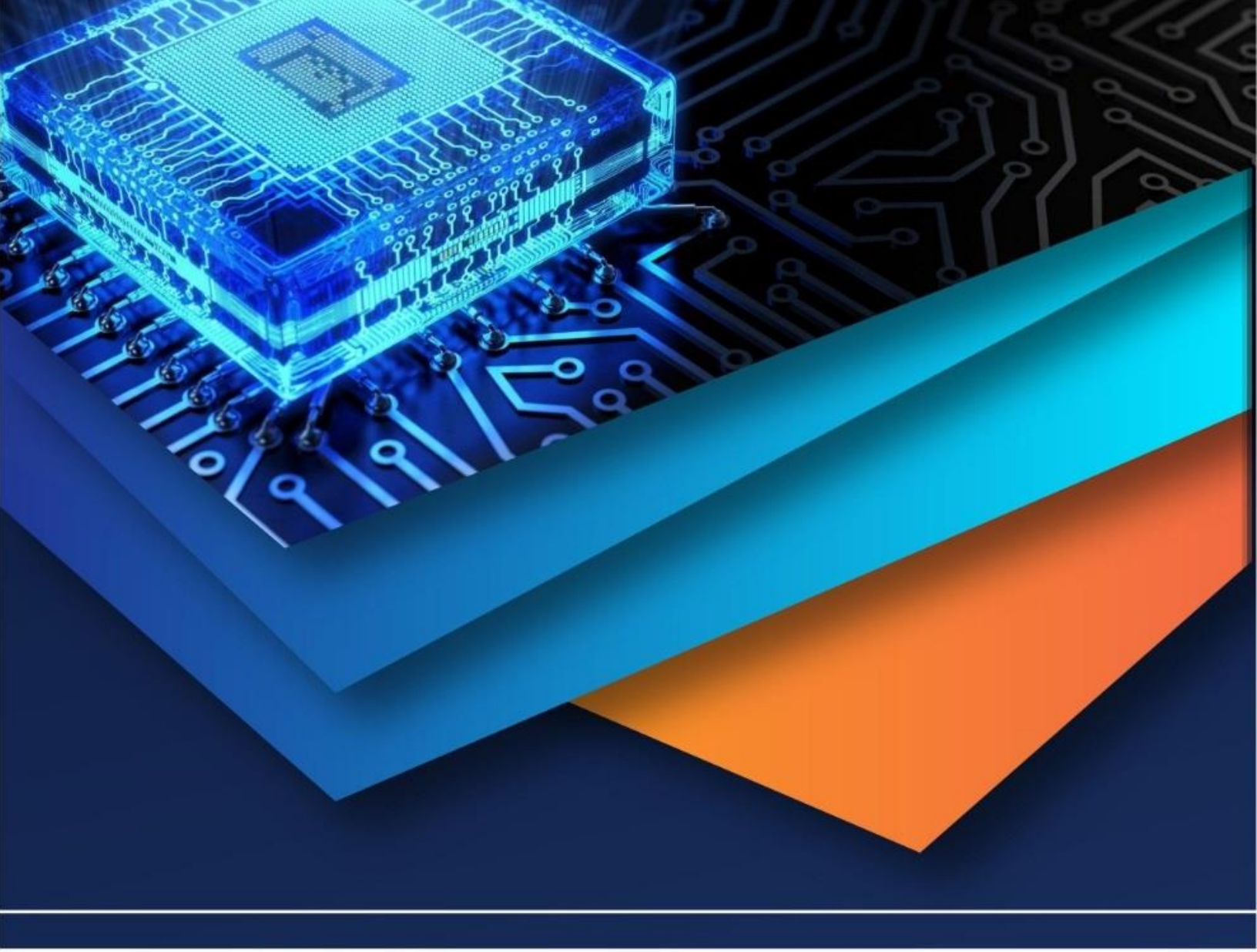
The performance in terms of makespan time for our proposed scheme is very enhanced. The energy consumption at same time also decrease without compromising with any add on time to makespan. The scheduling scheme utlize the computation power to maximum, as the result CPU is not idle anytime, which results in enhanceing the performance.

VI. CONCLUSIONS AND FUTURE WORK

In previous algorithm always largest task will be assigned to the best available resource (fastest) and does not consider the completion time. In our proposed scheme completion time is considered and it assigns the largest task to the slow VM and smaller tasks are in execution which improve the throughput. In this work the re-scheduling phase starts with finding resources makespan. The task with highest completion time and resource ($VM_{0)}$ is selected and there maximum completion time is compared with makespan for re-scheduling. When such task is found it is assigned to other virtual machine this will repeat until all the requests are not processed. Our results had shown significant improvement in makespan time and energy consumption.

REFERENCES

[1] Fox, R. Griffith et al., “Above the clouds: A berkeley view of cloud computing,” Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS, vol. 28, 2009.
[2] Foster, Y. Zhao, I. Raicu, and S. Lu, “Cloud computing and grid computing 360-degree compared,” in Grid Computing Environments Workshop, 2008. GCE’08. IEEE, 2008, pp. 1–10
[3] C. Germain-Renaud and O. F. Rana, “The convergence of clouds, grids, and autonomics,” Internet Computing, IEEE, vol. 13, no. 6, pp. 9–9, 2009
[4] Saeed Parsa, Reza Entezari- Maleki. 2009. RASA: A New Grid Task Scheduling Algorithm. International Journal of Digital Content Technology and its Applications. Volume 3, Number 4
[5] O M Elzeki, M Z Reshad and M A Elsoud. Article: Improved Max-Min Algorithm in Cloud Computing. International Journal of Computer Applications 50(12):22-27, July 2012. Published by Foundation of Computer Science, New York, USA
[6] Upendra Bhoi1, Purvi N. RamanuJ”Enhanced Max-min Task Scheduling Algorithm in Cloud Computing” International Journal of Application or Innovation in Engineering &Managenet (IJAIEM)” April 2013 ISSN 2319 – 4847
[7] Weiwei Chen; Deelman, E., "WorkflowSim: A toolkit for simulating scientific workflows in distributed environments," E-Science (e-Science), 2012 IEEE 8th International Conference on , vol., no., pp.1,8, 8-12 Oct. 20
[8] R. N. Calheiros, R. Ranjan,A. Beloglazov, C. A. F. De Rose, and R. Buyya. "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms", Software: Practice and Experience, 41(1): 23-50, Wiley, January 20
[9] Chieu T.C., Mohindra A., Karve A.A., Segal A., “Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment ,” in IEEE International Conference on e-Business Engineering, Dec. 2009, pp. 281-286.
[10] Naidila Sadashiv, S. M Dilip Kumar, “Cluster, Grid and Cloud Computing: A Detailed Comparison,” The 6th International Conference on Computer Science & Education (ICCSE 2011) August 3-5, 2011. SuperStar Virgo, Singapore, pp. 477- 482
[11] Vincent C. Emeakaroha, Ivona Brandic, Michael Maurer, Ivan Breskovic, “SLA-Aware Application Deployment and Resource Allocation in Clouds”, 35th IEEE Annual Computer Software and Application Conference Workshops, 2011, pp. 298-303
[12] V. C. Emeakaroha, I. Brandic, M. Maurer, and S. Dustdar, “Low level metrics to high level SLAs - LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in cloud environments,” In High Performance Computing and Simulation Conference, pages 48 – 55, Caen, France, 2010.
[13] M. Maurer, I. Brandic, V. C. Emeakaroha, and S. Dustdar, “Towards knowledge management in self-adaptable clouds,” In 4th International Workshop of Software Engineering for Adaptive Service-Oriented Systems (SEASS’10) , Miami, Florida, USA, 2010.
[14] Hagras and J. Janecek, “A high performance, low complexity algorithm for compile-time task scheduling in heterogeneous systems,” Parallel Computing, vol. 31, no. 7, pp. 653–670, 2005.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY