iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** VII **Month of publication:** July 2020

DOI: https://doi.org/10.22214/ijraset.2020.30683

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: 7.429
Volume 8 Issue VII July 2020- Available at www.ijraset.com

Enhanced Heuristic Cloudlet Allocation Based on Execution Time and Earliest Finish Time

Shilpa Manoharan[1], Shijin Knox G U[2]
[1, 2]*Dept. Information Technology Government Engineering College Trivandrum, Kerala, India*

Abstract: ***Cloud computing is an information technology paradigm that enables access to various shared pools of configurable system resources and higher level services. It provides delivery of computing services like storage, servers, database etc. Task scheduling is an important part in cloud computing for limited number of heterogeneous resources and increasing number of user tasks. In case of great cloudlets with high Size may lead fail in load balancing, increased makes pan and decreased average resource utilization. This paper provides an efficient algorithm to solve the above issues in task scheduling.***
Index Terms: HCA, EHCA, OCT, EDCB

I. INTRODUCTION

Cloud computing is the on-demand availability of computer system resources, especially data storage (cloud storage) and computing power, without direct active management by the user. The term is generally used to describe data centers available to many users over the Internet. Large clouds, pre-dominant today, often have functions distributed over multiple locations from central servers. If the connection to the user is relatively close, it may be designated an edge server. Clouds may be limited to a single organization (enterprise clouds), or be available to many organizations (public cloud).Cloud computing relies on sharing of resources to achieve coherence and economies of scale. Advocates of public and hybrid clouds note that cloud computing allows companies to avoid or minimize up-front IT infrastructure costs. Proponents also claim that cloud computing allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and that it enables IT teams to more rapidly adjust resources to meet fluctuating and unpredictable demand, providing the burst computing capability: high computing power at certain periods of peak demand. Cloud providers typically use a ”pay-as-you-go” model, which can lead to unexpected operating expenses if administrators are not familiarized with cloud-pricing models. The availability of high-capacity networks, low-cost computers and storage devices as well as the widespread adoption of hardware virtualization, service-oriented architecture and autonomic and utility computing has led to growth in cloud computing.

In recent years, cloud computing has emerged as heterogeneous distributed computing system to manage and allocate computing resources to user applications over the Internet in a self-service, dynamically scalable and metered manner[1]. User application can be divided in small size or high size of tasks (cloudlets). The advantage of high size is that the number of cloudlets is more reduced than with small size. However, a great number of large cloudlets whose cloudlet corresponds to high size lead the cloud system to fail in good load balancing and lower completion time. This is caused by their high execution time and delay during their execution on virtual machine (VM). Due to the security issues related to virtualization [7], all these VMs are distributed among increasing number of cloud users to complete running of their cloudlets. This means that the number of these VMs is much more reduced per single user

Optimizing task scheduling with good load balancing and minimum makespan which takes into account of large and small cloudlets under resources capacity constraints, such as number of VMs and processing speed, still remains the major challenging issue not well solved in cloud computing[1]. Therefore, task scheduling being an NP-hard problem, then a best optimal heuristic cloudlet allocation is required for managing high number of large and small cloudlets submitted by a single user to maximize both load balancing and resource utilization and to minimize overall completion time.

Based on task scheduling, many heuristic algorithms have been proposed to minimize the execution time of a task and finish time such as first strategy algorithm [1], round robin

(RR) [1] and standard deviation based modified cuckoo optimization algorithm (SDMCOA). FSA is based on a deadline, length of cloudlets and speed of execution of VMs, and vector that defines the number of cloudlets per VM, is distributed to each VM. FSA lacks in considering the completion time of VM and size of large cloudlets. As a result, this lack leads to higher completion time and improper load balancing. RR uses the ring as its queue to store cloudlets, and it allocates resources in circular order without using priority of the cloudlets. Each cloudlet in a queue has a same fixed unit of time, called quantum, allocated by scheduler and it

©IJRASET: All Rights are Reserved

will be executed in turn. If a cloudlet is unable to complete during its turn, it will be stored back to the queue waiting for the next turn. Large cloudlets are often assigned to the VMs with low MIPS (million instructions per second) increasing waiting time and overall completion time. As a metaheuristic method, SDMCOA allocates cloudlets to suitable virtual ma-chines by using fitness function based on finding maximum finish time. However, all these algorithms lack in finding a common completion time among heterogeneous VMs which may impact better on load balancing and overall completion time. In addition, there is no need to consider number of cloudlets, but the size of large and small cloudlets has to be considered. However, these heuristic algorithms do not focus on minimizing the completion time of each VM instead of only minimizing the execution time of a cloudlet which can increase the overall completion time.

A heuristic allocation algorithm has been proposed to solve the load imbalance problem based on the calculation of optimal completion time and earliest finish time[1]. But due to the distribution of cloudlets under this OCT, it fails in maximizing resource utilization and lead cloud system to fail in load balancing.

Here our work mainly focus on designing an enhanced heuristic cloudlet allocation algorithm for resource allocation and task scheduling, referred as EHCA to cope with the increasing large amount number of cloudlets submitted by a single user over heterogeneous virtual machines.

The proposed algorithm is divided in two phases, namely allocation based on execution time and allocation based on earliest finish time. In allocation based on execution time, the execution time of different task on each VM is calculated to ensure proper load balancing between all VMs. From global queue, user cloudlets are assigning to the VMs under the minimum execution time. In allocation based on earliest finish time, the unallocated user cloudlets are assigning to the VMs under earliest finish time in order to minimize waiting time, completion time of cloudlets and to maximize resource utilization

We simulates the proposed EHCA algorithm and the recent heuristic algorithm HCA. To measure the performance of the proposed algorithm, we compare the algorithm against the recent ones using various metrics such as degree of imbalance (DI), makespan and resource utilization (RU).

II. LITERATURE SURVEY

Many heuristics and meta-heuristics algorithm has been proposed to solve the load balancing problem in cloud computing[12]. But all of these algorithm lacks in considering the length of the task. Instead they are only focusing on the virtual machine's parameters. When considering min min algorithm[2],first it runs the small tasks and then later it executes the large tasks. So many tasks are to be in waiting stage and it causes the load imbalance problem. If we considers the length of task here, we can solve this issues. When it comes into max min algorithm[2], here first it runs the task with great length then after it takes the task with short length.

To improve task scheduling, many techniques based heuristic algorithms have been implemented such as heterogeneous earliest finish time (HEFT) [10] and minimum completion time (MCT) [15][6]. MCT assigns cloudlets in random order to the VM having earliest completion time. In this method, some tasks are allocated to the VMs without having minimum execution time. HEFT sorts firstly a list of cloudlets in decreasing order of upward rank calculated on basis of execution time and then, assigns a cloudlet to VM with earliest finish time using insertion-based approach.

Range wise busy-checking 2-way balanced (RB2B) also developes to solve the load balancing problem. This algorithm focus on distributing the number of cloudlets to the VMs in a uniform way with a balance threshold and earliest finish time. The RB2B algorithm considers the processing speed of each VM and cloudlet length range. But id does not consider the completion time of a VM. The length of the cloudlet will be proportional to the execution time of task. So it causes greatest completion time. So it also lead some degree of load balancing problem.

The recent heuristic algorithm[1] says that the calculation of optimal time can solve all the problems appeared on the min-min amd max-min algorithms. Here it first calculates the execution time of each cloudlet on each VM and then calculates the optimal completion time from the completion time of VM.Then it allocates the cloudlets under this optimal completion time at first phase. In the second phase it calculates the earliest finish time of the cloudlets which are not allocated in the first phase.

By doing this way, we can see that, in first phase, some of the VMs are not busy and it does not contain any task during first few seconds.

Thisnis because of the allocation under this optimal time. Also some Vms are running more number of tasks[1].So the main problem associated with this algorithm is some degree of load imbalance and decreased resource utilization. So our proposed work focus on solving all these problems associated with the heuristic allocation approach algorithm.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VII July 2020- Available at www.ijraset.com

III. PROPOSED SCHEME

Cloud Computing is emerging field for research and study. It is a pool of multiple configurable computing resources available on demand to user. The field is evolved from past technologies like web services, hardware virtualization, grid and utility computing, system management.

Moreover, Concepts of virtualization is used in cloud which leads to have a load balancing in the cloud. Virtualization means giving a logical name to physical resources and when-ever this name is referred it will point towards corresponding physical resource. Multiple users will access cloud at same time and it is very necessary to serve them all with minimum response time and better service. For this reason load balancing is taken in to effect to balance the request of multiple users on virtual machines evenly. It is said that Load balancing is a NP-Complete Problem method because as the size of the problem increases the size of solution will increase too. That means the more the request comes to cloud it will get tougher to do balancing amongst the Different Virtual Machines.

A. Load Balancing Operation

Load balancing is achieved in cloud environment in two steps: first one is to distribute the task among the node, second one is to monitor the virtual machine and perform the load balancing operation using task migration or virtual machine migration approach[9]. The aim of task scheduling is to create a schedule and assigned each task to node (virtual machine) for specific time period so that all task are executed in minimum time span. Task scheduling is NP complete problem in the field of computer science because number of task and length of task change very rapidly in cloud environment. It is difficult to calculate all possible task-resource mapping in cloud environment and find an optimal mapping is not easy task. Therefore we need an efficient task scheduling algorithm that can distribute the task in effective way so that less number of virtual machine should be in overloaded or under loaded condition. After allocating the task to virtual machine, cloud task scheduler start to perform load balancing operation so that task can be transfer from overloaded virtual machine to under loaded virtual machine and all virtual machine should remain in balance condition.

Before the allocation process, several parameters are to be considered for this scheduling algorithm.They are as follows; A virtual machine (VM) can be described as a tuple VM=id, mips, bw, pesnumber, where id represents iden-tifier of a VM, mips represents the processing speed per processing element (PE) at a VM, bw represents bandwidth of a VM and pesnumber represents the number of PE in a VM.

A cloudlet (C), referred to a task in this paper, can be described as tuple C =id, length, pesnumber, where id represents identifier of a C, length represents the size of C in MI (million instructions) and pesnumber represents the number of PE for running a C on a suitable VM.

A small cloudlet can be defined as a cloudlet whose length is small and needs a short execution time.

A large cloudlet can be defined as cloudlet whose length is large and needs a long execution time.A large cloudlet can be viewed as a set of dependent cloudlets with no communication cost. So here it will not be decomposed in multiple sequences of cloudlets to be executed on multiple virtual machines in order to minimize the com-munication cost at value of zero.

Assuming that there are 'm' VMs, and 'n' independent cloudlets (small or large cloudlets). Let eti,j be the execution time for cloudlet Cj corresponding to VMi , as shown by,

$$et_{ij} = \frac{C_j\ .length}{VM_i\ .pesnumber \times VM_i\ .mips}$$

where Cj .length denotes size of cloudlet Cj in MI, VMi .pesnumber denotes number of PE in a VMi , and VMi .mips denotes execution speed of VMi in MIPS. Thus, m VMs and n cloudlets will construct m × n cloudlet allocation matrix CA.

$$CA = \begin{bmatrix} et_{1,1} & et_{1,2} & et_{1,3} & \dots & \dots & \dots & et_{1,n-1} & et_{1,n} \\ et_{2,1} & et_{2,2} & et_{2,3} & \dots & \dots & \dots & et_{2,n-1} & et_{2,n} \\ \vdots & \vdots & \vdots & \dots & \dots & \dots & \vdots & \vdots \\ et_{m,1} & et_{m,2} & et_{m,3} & \dots & \dots & \dots & et_{m,n-1} & et_{m,n} \end{bmatrix}_{m \times n} \quad (2)$$

©IJRASET: All Rights are Reserved

In the below matrix CA, each row represents the execution time of different cloudlets processed on a targeted VM, and each column represents the execution time of a cloudlet on different VMs. The sum of execution time eti,j on each row i, defines the completion time in VMi

Let CTi denote completion time in targeted VM.It is defined by.

$$CT_i = \sum_{j=1}^{n} et_{i,j} \quad for\ i = 1, \ldots, m \qquad (3)$$

In order to reduce the execution time of each unallocated cloudlet, we define earliest finish time, denoted as EFT, at second phase of allocation of cloudlets.

Earliest start time (EST) is the earliest possible time that a cloudlet begins after the time that all predecessors are supposed to complete their execution. In our case, EST is initialized to CT determined at the first phase of the proposed algorithm

$$EFT_{i,j} = et_{i,j} + EST_i$$

Earliest finish time (EFT) is the earliest possible time that cloudlet can be finished from EST, as shown by 3.5. where ESTi is the earliest execution start time for cloudlet Cj on VMi.

B. Architectural Diagram

In our proposed architecture, virtual machines are created and allocated to the host(s) and are arranged in increasing order of processing speed. A cloudlet arrives from the global queue (GQ) to the effective datacenter broker (EDCB). EDCB is a cloud datacenter broker where the proposed EHCA algorithm is implemented. Fig.3.1 illustrates an overview of EHCA framework. The framework consists of three modules, as follows:

The first module is the user cloudlet which represents the set of the cloudlets.

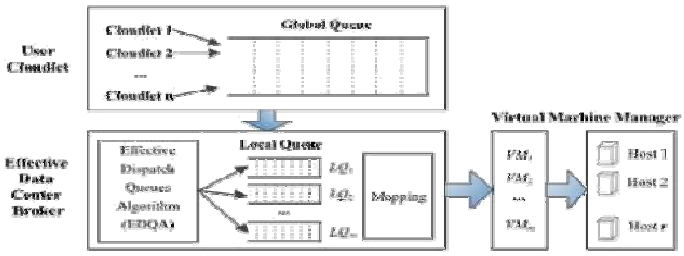


Fig. 1. Architectural diagram

The second module is effective data center broker which outputs a cloudlet local queue for each VM. In EDCB, there are two sub-modules which are as follows: effective dispatch queues algorithm (EDQA) and mapping. EDQA computes and finds all cloudlets for each VM according to optimal completion time and earliest finish time con-straints, while Mapping assigns each local queue to the corresponding VM.

The third module is virtual machine manager relying on different hosts. It manages a set of virtual machines that are used to execute user cloudlets. The cloudlets have been stored in an m×n matrix, where m is the number of virtual machines and n is the number of cloudlets.

C. Enhanced Heuristic Cloudlet Allocation Algorithm

EDQA targets a VM and selects a cloudlet which has minimum execution time in the targeted VM . Then, remove all allocated cloudlets from GQ

Second phase: Earliest Finish Time

EDQA checks whether all cloudlets from global queue are queued to be mapped on VMs, after completion of the first phase.

If the condition is satisfied, then terminate the second phase

EDQA assigns the non allocated cloudlets to VMs according to minimum EFT, until GQ becomes empty

©IJRASET: All Rights are Reserved

IV. EXPERIMENTAL SETUP

We have simulated the proposed scheme using CloudSim 3.0.3 simulator to evaluate the effectiveness of the proposed EHCA algorithm under different application scenarios[11] . All algorithms were written using Netbeans 8.2 Java program-ming language and running on a computer with Core i5-6500 CPU, 3.2GHz, 4G RAM.

A. Performance metrics

The performance metrics in our experiments are overall degree of imbalance (DI), average resource utilization (ARU), makespan and average waiting time of cloudlet (AWT).

DI is calculated by This algorithm is described in two phases, as follows: allocation based completion time phase and allocation based earliest finish time phase. At allocation based execution time phase, execution time of each cloudlet is calculated based on length of user cloudlets, resource capacity and number of heterogeneous virtual machines and a cloudlet which have less execution time are allocated to the corresponding virtual machine . Then the allocated cloudlets are removed from the global queue. At allocation based earliest finish time phase, earliest finish time based on execution time of each unallocated cloudlet and its earliest start time on a virtual machine is computed. Then the cloudlet with lowest EFT (earliest finish time) corresponding to a virtual machine is allocated to that virtual machine. This process continues until the number of cloudlets in the global queue becomes empty. By alloacting in this way, we can reduce the degree of imbalance and maximize the resource utilization. The algorithm for enhanced HCA algorithm is given below,

1) Input: A global queue of n cloudlets and list of m virtual machines
2) Output: Allocation of all cloudlets on suitable VMs
3) First Phase: Execution time

Arrange a set of virtual machines increasing order of processing speed Submit a list of cloudlets to EDCB
Calculate Execution time of different cloudlets on each VM.

$$DI = (T_{max} - T_{min})/T_{avg}$$

where Tmax , Tmin and Tavg are execution time maximum, execution time minimum and average total execution time of all VMs respectively

ARU is gaining significance as service providers want to earn maximum profit by renting limited number of resources. ARU is computed by

$$ARU = \frac{\sum_{i=1}^{m} CT_i}{makespan \times m}$$

where m is the number of allocated VMs and CTi , completion time described by fig 3.5.

Makespan is the maximum completion time of all VMs for running user cloudlets which is expressed as

$$makespan = \max_{1 \le i \le m} \{CT_i\}$$

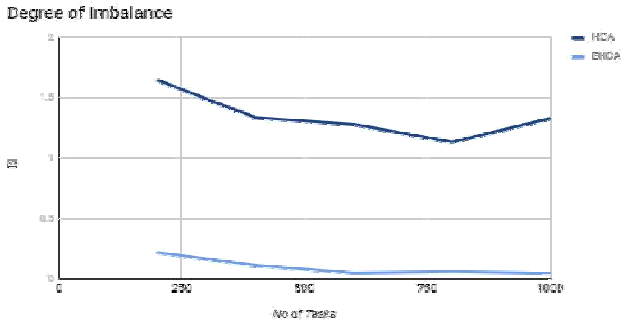


Fig. 2. Degree of Imbalance on short no of cloudlets

©IJRASET: All Rights are Reserved

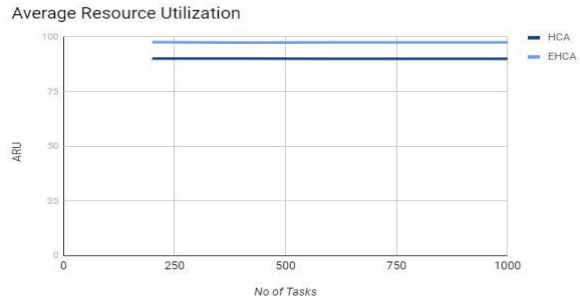


Fig. 3. Average Resource Utilization on short no of cloudlets

V. RESULT ANALYSIS

Here we are comparing the tasks which have short length and task which have high length on both EHCA and HCA algorithm.

A. Performance with Short Number of Cloudlets

To study the performance of our proposed EHCA algorithm without real workload for small cloudlets, the size of cloudlet is set between 100 MI and 1000 MI in random distribution executed on 30 heterogeneous VMs which are arranged in uniform distribution by increasing order and distributed among 4 hosts. The processing speed of VM is set between 500 MIPS and 10,000 MIPS.

1) *Degree of Imbalance:* The figure 5.1 shows the graph for the degree of imbalanace variation for the EHCA algorithm and the existing HCA algorithm. The x-axis shows the no of tasks and the y-axis shows the degree of imbalance. From the graph, it can be understand that the EHCA algorithm have minimum degree of imbalance.This is because many cloudlets are allocated to each VM with minimum value of execution time which depends on total length of all cloudlets.
2) *Average Resource Utilization:* The figure 5.2 shows the graph for the average resource utilization for the EHCA algorithm and the existing HCA algorithm. The x-axis shows the no of tasks and the y-axis shows the average resource utilization. From the graph, it can be understand that the EHCA algorithm has the maximum average resource utilization.

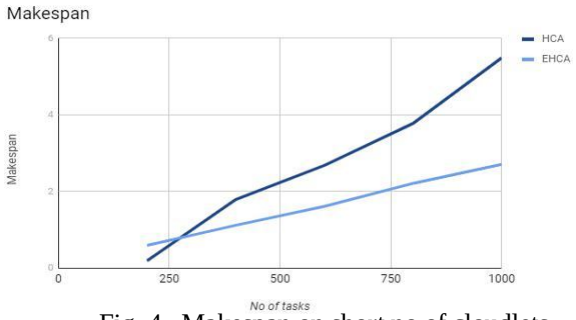


Fig. 4. Makespan on short no of cloudlets

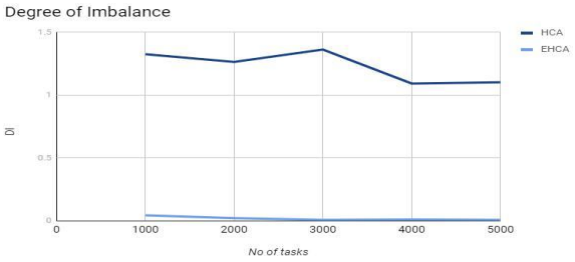


Fig. 5. Degree of Imbalance on large no of cloudlets

3) *Makespan:* The figure 5.3 shows the graph for the makespan for the EHCA algorithm and the HCA algorithm. The x-axis shows no of tasks and the y-axis shows the makespan. From the graph, it can be understand that the EHCA has minimum makespan.

©IJRASET: All Rights are Reserved

B. Performance with Large Number of Cloudlets

To study the performance impact of our proposed EHCA algorithm without real workload for large cloudlets, we set cloudlet size between 1000 MI and 2000 randomly; cloudlets are executed on 30 heterogeneous VMs which are arranged in uniform distribution by increasing order and distributed among 4 hosts. The processing speed of VM is set between 500 MIPS and 10,000 MIPS.

1) *Degree of Imbalance:* The figure 5.4 shows the graph for the degree of imbalanace variation for the EHCA algorithm and the existing HCA algorithm. The x-axis shows the no of tasks and the y-axis shows the degree of imbalance. From the graph, it can be understand that the EHCA algorithm have minimum degree of imbalance.This is because many cloudlets are allocated to each VM with minimum value of execution time which depends on total length of all cloudlets.

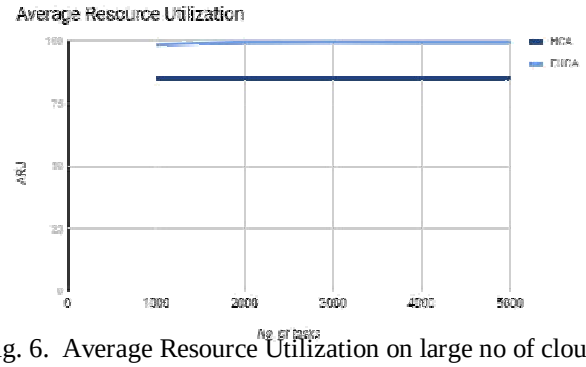


Fig. 6. Average Resource Utilization on large no of cloudlets

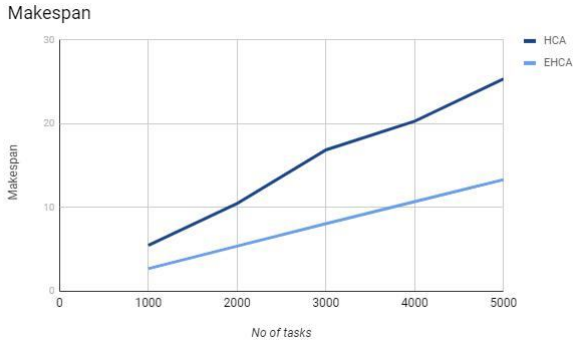


Fig. 7. Makespan on large no of cloudlets

2) *Average Resource Utilization:* The figure 5.5 shows the graph for the average resource utilization for the EHCA algorithm and the existing HCA algorithm. The x-axis shows the no of tasks and the y-axis shows the average resource utilization. From the graph, it can be understand that the EHCA algorithm has the maximum ARU.Resource utilization is far better maximized with EHCA than HCA due to achievement of load balancing based on ET and EFT.

3) *Makespan:* The figure 5.6 shows the graph for the makespan for the EHCA algorithm and the HCA algorithm. The x-axis shows no of tasks and the y-axis shows the makespan. From the graph, it can be understand that the EHCA has minimum makespan.

VI. CONCLUSION

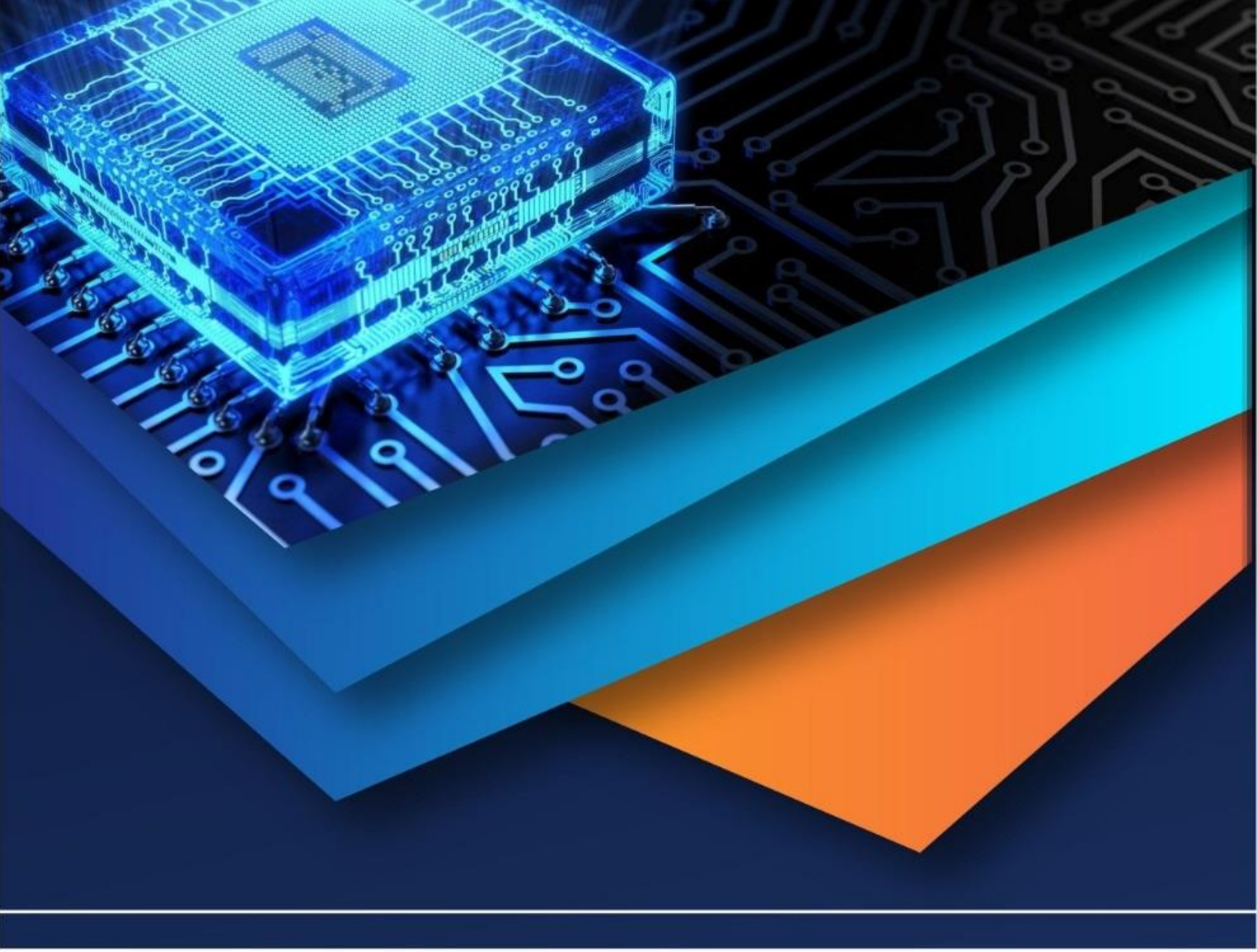
In this project, we proposed an enhanced heuristic cloudlet allocation approach (EHCA) which improves load balancing and task scheduling in cloud resource management under the environmental changing of the number of small cloudlets (tasks) and large cloudlets . The main innovation is the combination of minimum execution time and earliest finish time which implies the combination of execution time of cloudlets on each VM and earliest finish time based on execution time of a cloudlet. In addition, minimum execution time has been introduced to ensure a best load balancing while earliest finish time was for minimizing overall completion time and maximizing resource utilization. The enhanced HCA works into two phases that are as follows: allocation based on execution time and allocation based on earliest finish time. The detailed experimental results demonstrate that the enhanced HCA provides a best load balancing, highest resource utilization and lowest makespan in comparison with HCA algorithm. In the near future, we intend to extend our approach for live migration and security problem and on real cloud infrastructures.

©IJRASET: All Rights are Reserved

REFERENCES

[1] Jean Pepe Buanga Mapetu and Zhen Chen and Lingfu Kong (2018), "Heuristic Cloudlet Allocation Approach Based on Optimal Completion Time and Earliest Finish Time", International Journal of Pervasive Computing and Communications , 7:7–21.
[2]] R. J. Priyadarsini ,L. Arockiam, " Performance evaluation of minmin and max-min algorithms for job scheduling in federated cloud", Int. J. Comput. Appl, Volume 99, No 18, 2014.
[3] A. A. Buhussain, R. E. De Grande and . Boukerche (2016), "'Elas-ticity based scheduling heuristic algorithm for cloud environments", IEEE/ACM 20th Int. Symp. Distrib. Simulation Real Time Appl , 1-25.
[4] S. R. Shishira, A. Kandasamy and K. Chandrasekaran (2016), "Survey on meta heuristic optimization techniques in cloud computing", IEEE Int. Conf. Adv. Comput., Commun. Informat.,Q23.
[5] and H. Topcuoglu, S. Hariri and M. J. Usman (2002), "Performance-effective and low complexity task scheduling for heterogeneous comput-ing", ' IEEE Int. Conf. Adv. Comput., Commun. Informat,33:pp 775-93
[6]] S. Roy, S. Banerjee, K. R. Chowdhury, U. Biswas (2017) "'De-velopment and analysis of a three phase cloudlet allocation algorithm", "Development and analysis of a three phase cloudlet allocation algo-rithm," J. King Saud Univ. Comput. Inf. Sci., vol. 29, no. 4, pp. 473–483
[7] E. I. Djebbar, G. Belalem, "Tasks scheduling and resource allocation for high data management in scientific cloud computing environmen", Mobile, Secure, and Programmable Networking,(2016);35:1633–45.
[8] M. B. Gawali ,S. K. Shinde," 'Task scheduling and resource allocation in cloud computing using a heuristic approach", Int. J. Cloud Comput 2018.
[9] E. J. Ghomia, A. M. Rahmania, N. N. Qader, "Load-balancing algo-rithms in cloud computing: A survey", ' J. Netw. Comput. Appl(2017)
[10] R. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, R. Buyya,"CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algo-rithms", Softw., Pract. Expe,Volume 41 ,2011.
[11]] P. Humane, J. N. Varshapriya, "Simulation of cloud infrastructure using cloudsim simulator: A practical approach for researchers", Int. Conf. Smart Technol. Manage. Comput., Commun., Controls, Energy Mater, May 2015
[12] Q.-Y. Chen, Z.-H. Liang, H.-W. Kang, Y.-M. Ma,D. Wang: "'Re-search of dependent tasks scheduling algorithm in cloud computing environments" . c. 3rd Annu. Int. Conf. Inf. Technol. Appl" (2016).
[13]] R. Jithin, P. Chandran: "Virtual machine isolation: A survey on the security of virtual machines". . Int. Conf. Secur. Comput. Netw. Distrib. Syst, pp.. 91–102, 2014.
[14] D. Chaudhary, B. Kumar "An analysis of the load scheduling algo-rithms in the cloud computing environment: A survey". in c. 9th IEEE Int. Conf. Ind. Inf. Syst, 2014.
[15] S. H. H. Madni, M. S. A. Latiff, M. Abdullahi, S. M. Abdulhamid, M. J. Usman: "'Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environmen". PLoS One, (2017).
[16] H. B. Alla, S. B. Alla,A. Ezzati: "A novel architecture for task scheduling based on dynamic queues and particle swarm optimization in cloud computing". IEEE Int. Conf. Cloud Comput. Technol, 2016.
[17] R. K. Jena:"'Multi objective task scheduling in cloud environment using nested PSO framework". Int. Conf. Recent Trends Comput., 2015
[18] A. Khalili, S. M. Babamir:"'Makespan improvement of PSO-based dynamic scheduling in cloud environment". IEEE Iranian Conf. Elect. Eng., Tehran, Iran, May 2015, pp. 613–618.
[19] S. Singh, A. Ranjan:"'An improved task scheduling algorithm based on PSO for cloud computing". Int. J. Innov. Res. Comput. Commun. Eng., vol. 5, no. 11, pp. 16773–16777, 2017
[20] W. Hashem, H. Nashaat, and R. Rizk,:"Honey bee based load balancing in cloud computing" Internet Inf. Syst., vol. 11, no. 12, pp. 5694–5711, 2017.
[21] A. Beloglazov and R. Buyya:"Energy efficient resource management in virtualized cloud data centers" in Proc. 10th IEEE/ACM Int. Conf. Cluster, Cloud Grid Comput,2010
[22] Rodrigo N. Calheiros1, Rajiv Ranjan2, Anton Beloglazov1, Cesar A. F. De Rose, Rajkumar Buyya:"CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms" Cloud Computing and Distributed Systems,2010
[23] Tahani Aladwani:"Types of Task Scheduling Algorithms in Cloud Computing Environment" Open access peer-reviewed chapter
[24] Sellami, K., et al.:"Immune genetic algorithm for scheduling service workflows with QoS constraints in cloud computing" South African Journal of Industrial Engineering,2013,pp. 68-82.

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)