ISSN [Online]: 2583-2654

Energy efficient task migration and resource allocation using virtual machine pairing

R Prabhu [1] *and Dr S Rajesh* [2]

[1] *Assistant Professor, Department of Computer Science and Engineering, AAA College of Engineering and Technology, Sivakasi , Tamil nadu, India.*

[2] *Associate Professor, Department of Computer Science and Engineering, Mepco schlenk Engineering College Sivakasi Tamil nadu India.*

Abstract: Cloud computing has a rapid growth in the recent days. The high-performance and reliability paves greater way for the increase of cloud data centers. In the technical era, the most important aspect of cloud computing is virtualization and heterogeneous. However, cloud computing has greater popularity it has the same concern in exhibiting challenges mainly in real-time scenarios. One of the major issues still existing in a heterogeneous environment is load balancing. Task scheduling plays a vital role in balancing the load and achieving the overall system performances. Lack of QoS is a critical issue in the cloud system which is cause due to the increase of overloads and power consumptions. These issues can be effectively overcome by enabling optimization of resource utilization. Recently there are several mechanisms are evolved for balancing the load. Among which Virtual Machine (VM) migration technique is the most prominent way for achieving load balancing. The existing Virtual Machine (VM) migration technique has some limitations such as the overloaded VM cannot allow new tasks. Then the new task is handled by lightly loaded VM which increases the time consumptions and computational costs. In this paper, we propose (ACHR) adaptive VM migration technique using CHR (Communication History Record) system. According to which the proposed migration technique makes use of CHR during the task run time and migrates the task from overloaded VM to the next available VM. The CHR supports VM's previous and current status which enhances the resource mapping. According to which the VM placement strategy is improved and appropriate VM is allocated suitably to the task need. The task monitor allocation enhances the scheduling to execute within the deadlines. The proposed technique consists of task merging and VM pairing which enables task migration at shortest time with minimum consumption costs.

Keywords: Cloud computing, Task scheduling, and allocation, Load Balancing, Virtual Machines, Task Migration.

1. INTRODUCTION

Cloud computing is a boon of technology inventions. It enables a huge change in the aspect of virtual machines by performing migration, scheduling, consolidation and performing the tasks. The advantages of reliability and pay-per-use features are the valid reason for its increasing popularity. Cloud computing offers remote services where the user can use cloud services from anywhere and anytime. This launch cloud data centers in huge numbers and makes it's as a most profitable business. Generally, cloud computing avails three types of services such as IaaS

ISSN [Online]: 2583-2654

(Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service). By which IaaS provides required resource environmental for the requested users. SaaS provides software services and PaaS provides a platform for running the user applications. The on-demand computational service and minimum costs enable the user avoiding additional resource wastages [1]. The cloud computing framework is of distributed frameworks that increase the scalability. In the past decade's implementation of cloud computing has provided an ample number of new business opportunities commercially.

The cloud centers consist of several Physical Machines (PM) and each PMs are facilities their services to several users using the concept of VM (Virtual Machines). The single VM utilization or combined is of the user choice with their requirements. The computing resources of a PM are separated into multiple VMs built in the aspect to perform the large task promptly. The increasing workloads, task maximization, system efficiency makes VM more essential. The main challenge in this cloud computing using VM is load balancing. Load balancing may lead to failure of the task and increase the time along with the cost. This affects both QoS user experiences. To overcome various scheduling algorithms are evolved for balancing the cloud. These are developed on the motivation of mapping perfect virtual machines according to the user task needs. These are intended to balance the heterogeneity issues and the loads dynamically. Optimal resource utilization is achieved by allocating the VM which perfectly suits the user requirements. VM enables sharing among variable users. Each VM is different in the aspect of its memory size, CPU speed, and other resources. Even though VM are consider for solving load balancing but the resource underutilization problems needs to overcome [2]. The overloading issue is handled by implementing VM migration techniques. VM migration mechanism migrates the overloaded VM from one to another lightly loaded PM.

In this paper, we have proposed an adaptive VM migration technique using CHR (communication history record) which effectively migrates the overloaded VM from one to another. This paper is organized as follow section 1 consist of introduction. Section 2 describes the related work with the research issues and our proposed contribution. Section 4 describes detail about the proposed mechanism with its architecture and workflows. Section 5 has the results and discussion. The conclusion is given in section 6.

2. RELATED WORKS

Khoshkholghi (2017)developed Energy-Efficient Algorithms for merging dynamic virtual machines in cloud data centers. In this paper, the author addressed the aggressive VMs consolidation which may lead to performance degrades. The proposed work is aimed at maximizing resource utilization using SLA. The VM consolidation is overcome by enabling live migration which minimizes the energy consumption effectively.

Priya (2019) proposed an advanced resource scheduling algorithm for cloud services using **FUZZY**. In the cloud computing environment, scalable traffic management is implemented for balancing the load traffic. But still, it suffers from a lack of multidimensional resource allocation issues. The main intention of this proposed work is to implement integrated resource scheduling and load balancing algorithms for achieving cloud service provisioning. Itcan be done using the Fuzzy-based

ISSN [Online]: 2583-2654

Multidimensional Resource Scheduling model (FMRS). FMRS works effectively on dealing with complex queries, hence resource underutilization and overutilization issues are overcome effectively.

Latiff (2018) analyzed the task execution failure in cloud computing. The author addressed most of the scheduling algorithms are focused on the scheduling issues but considered fault tolerance. For this dynamic clustering league championship algorithm (DCLCA) scheduling mechanism is developed for handling fault tolerances. The DCLCA failure rate is very low in comparison with other conventional scheduling algorithms.

Sivagami (2019) researched the data computation issues that occurred during the grouping of virtual machines. The main issue of implementing the VM for migration is the number of VM required to allocate for migrating and how to maintain them effectively. To handle this new Dynamic Fault Tolerant VM Migration (DFTM) is proposed for cloud data centers. It applies the enhanced VN recovery mechanism for balancing the faults which maximize the VM survivability widely.

Agarwal (2018)state in his research that the maximization of cross- the virtual machine will lead to side-channel attacks and other security issues. These attacks may have the possibility of stealing user information and leads to severe privacy breach in the cloud environment. To avoid this author proposed a modern VM placement algorithm known as Previously Co-Located Users the First algorithm. It minimizes the malicious co-location with slight negotiation in resource utilization.

Pradeep (2018) proposed OCSA:a task scheduling algorithm for cloud environments. In this work, the author considers the resource optimization problem using makespan and cost. For which a new hybrid algorithmis evolved by combining the cuckoo search algorithm (CSA) and oppositional based learning (OBL) algorithms known as the oppositional cuckoo search algorithm (OCSA). The implementation of OCSA is more effective in handling dynamic resource utilization using the minimizing cost and makespan parameters.

2.1 Problem Statements

Cloud computing is very popular for its enhanced features and in the same aspect its concern towards the issues. The above-related works describe several issues based on scheduling, security, and performance along with the proposed methodology developed to manage it. In this work we have not discussed all the cloud-related issues, we analyze that optimal resource utilization is the prominent solution for enhancing the overall system performance. It can be achieved by implementing VM migration techniques. Some of the existing work has developed with VM migration but faces certain issues such as;

- Does not migrate the overloaded VMs properly
- Lack of identifying the best VM for the task
- Lack of identifying the suitable task which is running inside the current VM and migrate to the exact available suitable VM in case of overloads [12, 13].
- Increase of dirty memory during migration
- Increase of possibility of losing user information due to improper migration
- Lack of VM placement strategy such as how many VM is required and proper allocation according to the requirements.

2.2 Research methodology

Swanirman Sunirmit
Explore Technology advancements for Sustainable Ecosystem on Society

ISSN [Online]: 2583-2654

Our proposed works contributions as follows;

1) Achieving optimal resource utilization, in cloud services load balancing is an important issue that leads to a huge amount of power wastage. To save this power virtual migration technique is implemented. The existing works [12] in the proposed model VM migration is enhanced in identifying which are the VM's are available to compute. The extra task from the overloaded VM is migrated to the available resources.

2) The proposed mechanism consists of CHR-Communication history records. This CHR maintains the complete details about all the VM's performance which are prominent for effective VM migration. Its details working mechanism is explained in the proposed section.

3) In the system, once the running VM gets overloaded the new task does not allow until it gets completed. Instead, the overloaded is automatically find the available VM without any delay. In the case of handling the heavier tasks, task migration is applicable eventually which overcomes the delay issues in the mechanism

4) Task monitoring and scheduling with the monitored details from CHR improves the overall system efficiency.

5) To enable reliable user experience task queue is maintained by which the task request is taken from various users and queued based on FIFO.

6) For achieving quick task execution, merger is implemented for merging the queue and VM pairing is done using Dynamic task scheduler. It enables quick execution and acknowledges that all the allocated tasks get computed

7) By the proposed VM migration technique the next available VM is identified quickly and herewith the communication cost is reduced. Such a mechanism paves a way for task computation at minimum time.

3. PROPOSED METHODOLOGY

Fig 1 (last page of this article)

3.1 Proposed Architecture

The main motto of our proposed mechanism is the minimization of communication costs between the host and VM using VM migration. Additionally, our proposed work aimed to balance the overloads including resource optimization and improving overall system performances. By which the physical machine will intricate with the host and VMs even in the idle state. Most of the works state that the best way to reduce the overhead is by switching off physical machines during idle [20, 21]. The most successful way for achieving effective VM allocation is by implementing clustering techniques. The working mechanism of VM migration is built with PMs and VMs distance, fitness and similarity [21]. Our proposed mechanism also enhances dynamic resource monitoring and allocations. The QoS is depended on the effectiveness of dynamic resource monitoring and allocations. Initially, the monitoring is done with the cloud service providers who availing the physical resources like memory, network bandwidth, and CPU. Simultaneously the required resources needed for the task are also analyzed. The VM placement strategy works according to it and processes the scheduling.

ISSN [Online]: 2583-2654

The proposed architecture consists of a job request function, it collects the user request from the various location with various needs. The request may be of software, hardware or networking resources [21, 22].

Next, the collected job request is accepted by the monitoring section and verifies the user authentications. From a valid user, the request accepted and sent to the VM scheduler controller. The request from the malicious user is denied and reports are generated according to it. The dynamic environment makes resource scheduling as a challenging task [6, 7]. The user's job request may vary with time and application since it is dynamic. For handling the dynamic challenge controller took the responsibility for mapping the appropriate available resources according to the need. The Physical & virtual Resource monitoring section gives detail information about the available resources and this status (in the proposed mechanism it's done by CHR). On preparing the priority-based scheduling queue user's requirement, QoS, completion are considered. In our proposed scheme the dynamic task scheduling along with load balancing is done by VM migration with historic communication report provided by CHR. The monitoring is done regularly and let's discuss the proposed workflow as below.

Initially, the task request is collected from the various resources and collected in the queue for verifying the authentication. After verifying the authentication valid user task set are expressed as;

$$Tasks\ (t_{ix}) \rightarrow T_1, T_2, T_3 \ldots T_{ix} \quad \ldots. (1)$$

Where $T_{ix}$is the set of the task;

Next, the prepared task queue is sent to the task monitor section. Where the task monitor analyzes the requirement and finds the similarity to perform quickly. Once the task requirements are analyzed the available VM resource is a monitor to schedule the exact the resource as per the task. The VM placement strategy is done with CHR- communication history report. This CHR gives a detail report about the available VM present, previous status report. This report helps find the exact VM for the task. This can be done by;

The available virtual machines and host are $virtual\ machines\ (vm_{ix}) \rightarrow vm_1, vm_2, vm_3, \ldots. vm_{ix}$ and $host\ (h_{ix}) \rightarrow h_{ix}, h_{ix2}, h_{ix3} \ldots h\_ix_n$. Then the tasks are assigned in the queue$q_{ix} \rightarrow T_1, T_2, T_3 \ldots T_{ix}$. The task monitors collect VM's communication history record based on the parameters like speed, memory, bandwidth and data rate from CHR which can be expressed as;

$$T_{mointor} \rightarrow CHR \ldots (2)$$

Then the task monitor examined the available VM details are resulted as;

$$T_{mointor} \rightarrow vm\ details; \ldots (3)$$

Then the task monitor undergoes examining the task based on its size, length, etc. It differs according to the inputs such as for video resolutions, frames, width, and height are considered. For images, its pixel size, rows, and columns are considered. For sounds, its frequency, bit rate, file size are examined and the final result is expressed as;

$$T_{mointor} \rightarrow, t_1(t_s) \ldots (4)$$

$$T_{mointor} \rightarrow \\ claculate\ the\ length\ of\ the\ task\ size\ of\ (t_{ix_1}, t_{ix_2}, t_{ix_3} \ldots t_{ix_n}) \\ \ldots (5)$$

Then initialization of task allocation and monitoring process is done. The challenging issue is tasks differ

ISSN [Online]: 2583-2654

from time to time and as per the requirement. This dynamic task allocation is handled by task allocation and implementing task monitoring. The task allocation is done the basic resources, VMs, and tasks respectively.

$R_{allocate} \rightarrow based\ on\ cpu$;
$R_{allocate} \rightarrow vm_1(t_1), vm_2(t_2), vm_3(t_3) \ldots vm_n$;
$R_{allocate} \rightarrow T_{mointor}(vm) details$;
$R_{allocate} \rightarrow T_1(vm_1), T_2(vm_2), T_3(vm_3) \ldots . T_n(vm_n)$;

The monitoring is according to the data accessing time and completion time such as;

$$R_{mointor} \rightarrow \frac{d_t * t_t}{e_{ct}} \ldots . \quad (6)$$

Then the task successfully executed on the available resource. If the task running time is too long then the task is to migrate into the next available VM using the resource allocation function such as;

$$R_{mointor} \rightarrow t_{ix}(vm_{ix}) \sum_{t=i}^{n} d_t * t_r \ldots (7)$$

This mechanism is continued for executing the entire task in the queue till attaining successful completion.

Dynamic task scheduling migration using Adaptive communication record (ACHR)

1. $Input: set\ of\ task\ (T_{ix})\ and\ virtual\ machine\ (Vm_{ix})\ host\ (h_{ix})$;Output: throughput, efficiency,
2. Begin
3. {
4. Get (); data from the user // date set given the input;
5. $Tasks\ (t_{ix}) \rightarrow T_1, T_2, T_3 \ldots T_{ix}$;
6. Create the virtual machine and host
7. $virtual\ mechines\ (vm_{ix}) \rightarrow vm_1, vm_2, vm_3, \ldots . vm_{ix}$;
$host\ (h_{ix}) \rightarrow h_{ix}, h_{ix2}, h_{ix3} \ldots h\_ix_n$
8. Task assign in queue process
9. $q_{ix} \rightarrow T_1, T_2, T_3 \ldots T_{ix}$;
10. Monitor the communication performance through task with vm's performance
11. $T_{mointor} \rightarrow CHR, Vm's$; // CHR maintain the communication performances history records with vm's (memory, speed, bandwidth, data rate);
12. $T_{mointor} \rightarrow vm\ details$;
13. $T_{mointor} \rightarrow, t_l(t_s)$ // calculate task length and size for video (resolutions, frames, width and height), image (pixel size, rows and columns), sound (frequency, bit rate, file size)
14. $T_{mointor} \rightarrow$ $claculate\ the\ length\ of\ the\ task\ size\ of\ (t_{ix1}, t_{ix2}, t_{ix3} \ldots t_{ix_n})$ // monitor the performance task and vm's performance;
15. Initialize the resource allocation and monitoring
16. $R_{allocate} \rightarrow$ $based\ on\ cpu\ (memory, bandwidth, speed, data\ rate)$;
17. $R_{allocate} \rightarrow vm_1(t_1), vm_2(t_2), vm_3(t_3) \ldots vm_n$; // calculate the task;
18. $R_{allocate} \rightarrow T_{mointor}(vm) details$;
19. $R_{allocate} \rightarrow T_1(vm_1), T_2(vm_2), T_3(vm_3) \ldots . T_n(vm_n)$;
$R_{mointor} \rightarrow \frac{d_t * t_t}{e_{ct}}$ // data access time, task completion time / running time;
20. Case (i);
21. $R_{mointor} \rightarrow t_{ix}(vm_{ix}) \sum_{t=i}^{n} d_t * t_r$ // date access time, running time is too long migrate to another vm based on $R_{allocate} \rightarrow T_{mointor}(vm) details$;
22. Case (ii)
23. $R_{mointor} \rightarrow t_{ix}(vm_{ix}) \sum_{t=i}^{n} d_t * t_r$ // size is the same the task executed successfully
24. End case;
25. End

ISSN [Online]: 2583-2654

3.2 Scheduling Model

In this model, the scheduling system consists of CHR (Communication Historical Records), VM's details, resource allocator and monitor. Initially, the user sends the task request and request is collected in the queue in and sorted descending to deadlines. On the arrival of the task, the task observer segregates the task and allocates the appropriate VM of its type. The task execution is performed as per the proposed algorithms which are described in the below sections. The final decision from the task observers and details from CHR determines the merging queue. If the task is of new type then-new VM of its type is created. The available VM faces performance laciness due to a shortage of resource requirements such as Disk Storage, Network Bandwidth, RAM and CPU then the resources are scale-up by the resource allocator. The hosts performing ability and variation among first and second tasks are checked by the resource monitors. If it's finding to be a similar type then the task merger process the merging queue. According to the execution status, task scheduling and resource utilization are schedulers directly. The resource monitor regularly monitors physical hosts status and the resource allocator works according to its. The VM migration or resource boosting is decided by the resource allocator. The fig1 shows the system model in detail.

The user requirement in a medium or container is known as a task. Generally, tasks are of two types such as task computing with high CPU memory and second is task execution requires low CPU memory. The task set can be described as $TTasks\ (t_{ix}) \rightarrow T_1, T_2, T_3 \ldots T_{ix}$. Each task set consists of task parameters such as primary memory, disk storage, processing capacity, and network bandwidth. The CHR consist of historical task details, type count and task type.

3.3 Scheduling of Virtual Machine

The process of scheduling virtual machines is similar to a single cloud environment with various data globally. The scheduling algorithm took the main responsibility of scheduling optimization process with available information. In this infrastructure, the VMis deployed when workloads are transferred from one cloud infrastructure to another infrastructure. The scheduling process of VM includes the mapping of virtual machines and physical machines. Physical machines are sorted under the hosts. The challenging part of the VM scheduling process is allocating enough resources for large applications. In these cases, each larger task is split into subtasks. For every subtask single, VM instance is requested and allocated. For instance set of task $Tasks\ (t_{ix}) \rightarrow T_1, T_2, T_3 \ldots T_{ix}$ to the VMs' $virtual\ machines\ (vm_{ix}) \rightarrow vm_1, vm_2, vm_3, \ldots. vm_{ix}$ and $host\ (h_{ix}) \rightarrow h_{ix}, h_{ix2}, h_{ix3} \ldots h\_ix_n$.

3.4 Virtual machine scheduling predictions

The VM's placement problem is overcome by enabling each server or host with a virtual machine for a particular time. It confirms the required resources to compute the workload successfully. However, it does achieve resource optimization because the pattern of workload may vary. This needs a single VM which satisfies multiple requirements and improving the overall resource utilization. The major consideration is needed to give the workload fluctuation while predicting the VMs [14]. For an effective VM placement, an analysis report which includes past and current VM utilization is required. This

ISSN [Online]: 2583-2654

can be obtained in our proposed scheme by implementing CHR (communication history records). This gives a stable and detailed report about the VM's idle or active status. It is well applicable for handling a new task to place appropriate VM. The prediction can be improved with accuracy by implementing machine learning on the predictive algorithm for resource utilization. The VM behaviors and CPU utilization are well trained by using an artificial neural network [15].

3.5 Virtual Machine Migration

The major and existing challenge in the data center is properly managing and scheduling of virtual machines. This can be done to various users who have different requirements of several kinds. Server failure, overloading issues are also the reason for affecting overall performances. The only available solution is implementing VM migration and resource distribution with optimal utilization. The main parameters required for proper VM migration are the necessary availability of CPU, memory, etc. In VM migration, a server is not provided with a dedicated VM or vice versa. These VM's are migrated from server to server according to the requirements [16]. VM migration manages downtime resources maintenance and improves data center performance efficiencies [17]. On VM migration the important thing to be noted is connection and network continuity till the end. Virtual Machine migration is of VMs live and non-live state with memory data. These memory data are stored and transferred, on the case of any interruption or failure occurs these details helpful in continuity. The performance of Virtual Machine migration is determined using the evaluation matrices such as migration times between the servers, its network consumption, in case of non-live VM migration its downtime and time consumption while transferring the data.

4. RESULTS AND DISCUSSION

In this research work, the performance of our proposed ACHR is evaluated by conducting experimental work in cloudsim tool. The cloudsim tool is a simulator that enables easy modeling and functionalities. The simulation is the best way to contribute to the research work which is proven by the previous working using this tool. This cloudsim is easy to implement and analyze. The visualization representation makes understanding clear and in a simple manner. In this work data center work model is deployed with VM and hosts. These are implemented according to the host and VM implements policies. The comparison work is carried between our proposed ACHR with EEA and FMRS. The analysis is based on three terms such as Total mean response time with task count, Resource utilization with task count and Waiting time with task count. The experiment is conducted with 10 hosts and 10 VM's respectively. The simulation time is examined in a sec with several aspects as mentioned above. Below table 1 represents the different parameters taken for the experiments.

Table1: Simulation results from cloud sim

4.1 Mean Response Time

The total mean response time is referred to the actual time taken for the response. The minimum time states the level of efficiency. The mean response time is calculated with different task types with different counts. The improved VM placement strategy in our proposed ACHR reflects an effective response as there no delay due to the implementation of the CHR system. The comparison result between the three algorithms is given below.

Swanirman Sunirmit
Explore Technology advancements for Sustainable Ecosystem of Society

ISSN [Online]: 2583-2654

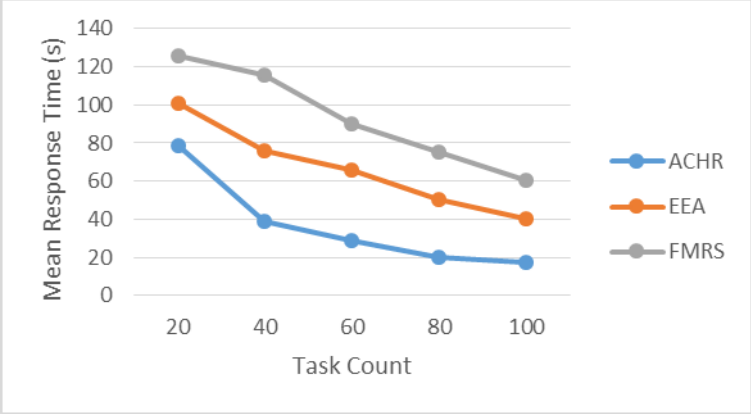


Fig 2: Mean response time VS Task count

The above fig 2 describes the performance evaluation based on mean response time. The total meantime is calculated in sec. It holds the user request submitted analysis and response with VM allocation. The X-axis determines the total mean response time and the Y-axis states the task counts. The obtained observations are separated by three different colors respective to their algorithms. Time taken for the count of 5 tasks among three algorithms is ACHR>EEA>FMRS such as. In the same manner for every increase of load observation is plotted and represented in the above graph. From the above figure, it clears that proposed ACHR achieves response in minimum time in comparison to EEA and FMRS respectively.

4.2 Resource Utilization

Optimum resource utilization is the best way to achieve efficiency. Maximum resource utilization improves the overall system performance with minimum time and cost. This experiment states about the total resource utilization with multiple tasks.

4.3 Average success rate

In cloud computing, average success rate is the measurement of VM's idle without running a task. This may happen due to improper task scheduling as the VM needs to get the allocated task for execution. The maximum average success rate leverage the performance and results in QoS issues. The comparison waiting time between the three algorithms are given below.

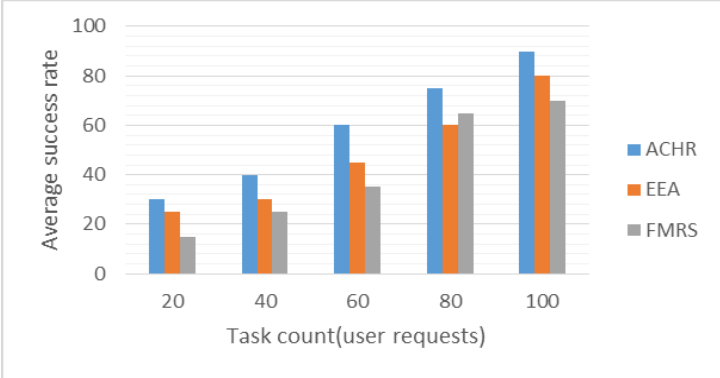


Fig 4: Average success rate VS Task count

The above fig 4 is the graphical representation of the average success rate obtained by each algorithm respective to different task loads. The X-axis determines the total waiting time and the Y-axis states the task counts. The total waiting time is calculated in sec. The average success rate obtained for the count of 5 tasks among three algorithms is ACHR>EEA>FMRS. The obtained observations are separated by three different colors respective to their algorithms. It is cleared on comparison the proposed ACHR consumes minimum waiting on all stages than the others.

5. CONCLUSION

This paper demonstrates detailed analysis of various scheduling techniques and their impact on Cloud computing. User experience and QoS are the two major aspects of cloud computing. It can efficiently be achieved by proper virtualization and heterogeneous mechanism. From the analysis of recent works, it is clear load balancing issues can overcome by implementing VM migration techniques. But still,

ISSN [Online]: 2583-2654

VM idle state and denial of new task on overloaded VM are the major challenge exists. This was overcome in our proposed work by implementing adequate VM migration using the CHR system. In this work, the major role is played by CHR which gives complete communication history detail about the VM. By which the allocation of appropriate VM to the task is done as per the requirements. In case of overload, automatically the task from the current VM is migrated to the next available VM. The CHR supports VM's previous and current status which enhances the resource mapping exact. To evaluate the performance of ACHR a comparison work is carried with EEA and FMRS. The comparison results show on all the evaluation parameters ACHR achieves effective results than others. Thus proves our ACHR is the advanced VM migration technique that overcomes the existing challenges in a reliable manner.

REFERENCES

[1] Adhikari, M., &Amgoth, T. (2019 Deadline-Aware Scheduling for Scientific Workflows in IaaS Cloud. Smart Innovations in Communication and Computational Sciences (pp. 347-360). Springer, Singapore.

[2] Basu, S., &Anand, A. (2019). Location Based Secured Task Scheduling in Cloud. In Information and Communication Technology for Intelligent Systems (pp. 61-69). Springer, Singapore

[3] M. A. Khoshkholghi, M. N. Derahman, A. Abdullah, S. Subramaniam, and M. Othman, "Energy-efficient algorithms for dynamic virtual machine consolidation in cloud data centers," IEEE Access, vol. 5, pp. 10 709–10 722, 2017.

[4] Priya, V., Kumar, C. S., &Kannan, R. (2019). Resource scheduling algorithm with load balancing for cloud service provisioning. Applied Soft Computing, 76, 416-424.

[5] Latiff, M. S. A., Madni, S. H. H., &Abdullahi, M. (2018). Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm. Neural Computing and Applications, 29(1), 279-293.

[6] Sivagami, V. M., &Easwarakumar, K. S. (2019).An Improved Dynamic Fault Tolerant Management Algorithm during VM migration in Cloud Data Center.Future Generation Computer Systems, 98, 35- 43.

[7] Agarwal, A., & Duong, T. N. B. (2018, December).Co-Location Resistant Virtual Machine Placement in Cloud Data Centers.In 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS) (pp. 61-68).IEEE.

[8] Krishnadoss, P., & Jacob, P. (2018).OCSA: task scheduling algorithm in cloud computing environment.Int J IntellEngSyst, 11(3), 271-279. and Systems, Vol.11, No.5, 2018 DOI: 10.22266/ijies2018.1031.25

[9] Abazari, F., Analoui, M., Takabi, H., & Fu, S. (2019). MOWS: multi-objective workflow scheduling in cloud computing based on heuristic algorithm. Simulation Modelling Practice and Theory, 93, 119-132.

10] S. Supreeth and KiranKumariPatil , "Virtual Machine Scheduling Strategies in Cloud Computing- A Review", International Journal on Emerging Technologies 10(3): 181-188(2019).

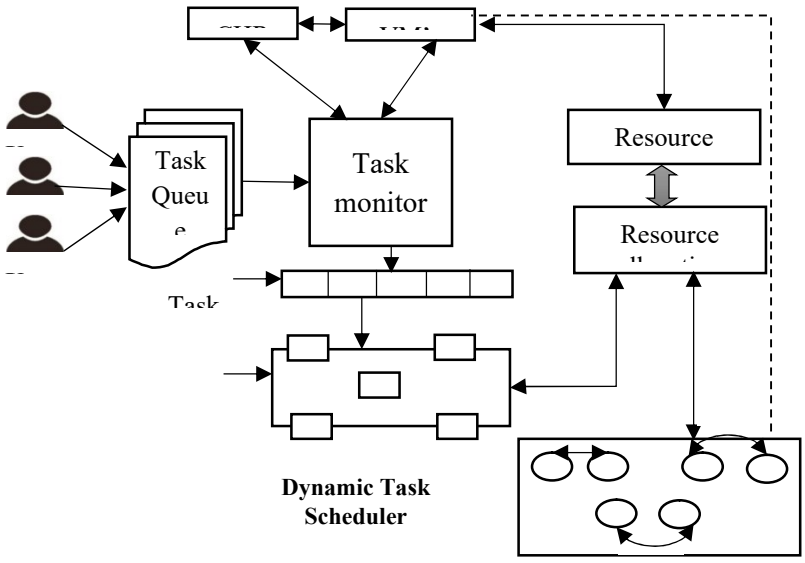
[11] F. Ramezani, J. Lu, and F. K. Hussain, "Task-Based System Load Balancing in Cloud Computing Using Particle Swarm Optimization", International Journal of Parallel Programming, Vol. 42, No. 5, pp. 739-754, 2014.

[12] N. Jain, I. Menache, J. Naor, and F. Shepherd, "Topology Aware VM Migration in Bandwidth Oversubscribed Datacenter Networks", In: Proc. of the 39th International Colloquium, pp. 586-597, 2012.

Swanirman Sunirmit
Science, Technology and Engineering for Sustainable Ecosystem or Society

ISSN [Online]: 2583-2654

Fig 1



Task ID	STATUS	Data center ID	VM ID	Arrival Time	Exec Time	Finish Time	Dead line
0	SUCCESS	2	9	1.00	17.89	17.89	0
1	SUCCESS	2	9	364927.48	364928.48	364928.48	1
2	SUCCESS	2	5	626990.24	628050.21	644538.04	2
3	SUCCESS	2	3	879172.62	879612.35	880262.02	3
4	SUCCESS	2	2	1135539.045	1135544.04	1135544.044	4

Table1: Simulation results from cloud sim