Design of a Fault Tolerant Strategy for Resource Scheduling in Cloud Environment

Keerthika P, Suresh P, Manjula Devi R, Sangeetha M, Sagana C

Abstract: Cloud computing supports the technological need of the industry supporting many other technologies. Also, the demand for computing power and storage by recent technologies is reasonably growing in a drastic way. Cloud computing, serving for these technologies are to be developed with advancements that lead to performance improvement both in support to the technologies like block-chain and big data. The allocation of cloud resources is an important strategy to be followed in a wiser manner to incorporate the needs of extra ordinary computing power. In this paper, an efficient resource allocation strategy (FTVMA) is introduced that involves the creation of effective virtual machines (VMs) and performs VM allocation in an efficient manner by considering the failure rates, previous history of failure of VM, execution efficiency as a part of effective scheduling. There exist many reasons for cloudlet failure in VMs. Some of them are overloading of VMs and non-availability of VMs. The introduced FTVMA algorithm considers the failure rate of the physical machine, load of virtual machines and the cost priority of the tasks in order to achieve Quality of Service (QoS) and Quality of Experience (QoE) of the user. The FTVMA methodology proposed in this paper works better for computation intensive VMs and is tested using CloudSim environment. The QoS metrics used to measure the performance of the proposed algorithm are Makespan and VM Utilization. The metric to measure QoE are Priority Miss Rate and Failure Rate. The proposed algorithm shows its improvement in terms of the QoS and QoE metrics. The results obtained are compared with the existing resource scheduling algorithms and it is inferred that the proposed algorithm performs better in terms of QoS and QoE.

Keywords : Cloud computing, Fault tolerant, Load balancing, Scheduling, VM allocation.

I. INTRODUCTION

Cloud computing is an emerging computing technology which is nowadays used by many of the other technologies such as big data, block-chain technology, deep learning and wherever there is a need for processing large amount of data. Worthwhile, the amount of data stored in cloud are enormous and they are likely to be secured. In general, resource sharing is the main objective of cloud. The resources such as memory, CPU, infrastructure, networks, bandwidth etc., are being shared via cloud. The resource providers are basically responsible for the effective utilization of these resources and also the security of the data. The National Institute of Standards and Technology (NIST), defines the term "cloud computing" as:

"A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [12].

Cloud computing is delivering softwares, platform and infrastructure as services. These services are offered to the consumers in a pay-as-you-go model. In industrial terms, offering software is referred as Software as a Service (SaaS), offering platforms is referred as Platform as a Service (PaaS) and offering infrastructure is referred as Infrastructure as a Service (IaaS). Many computing service providers including Google, Microsoft, Yahoo and IBM are rapidly deploying data centers in various locations around the world to deliver Cloud computing services [2].

Nowadays, cloud to cloud backup is trending in the cloud computing era. Cloud-to-cloud backup is a process of backing up a copy of data stored in one cloud to another cloud. The cloudlets submitted by the end users should be executed by the cloud VMs which are associated with the host machine. In cloud environment, satisfying the users, require attention on several areas like Scheduling, VM consolidation, VM migration, etc. Many of the cloud computing infrastructures consist of reliable services delivered through knowledge centers. They in turn are working on hosting a huge variety of applications starting from tasks that endure less time to those tasks that endure longer time for completion. Once the employment in terms of tasks is submitted to the clouds, it's divided into many cloudlets [3].

Scheduling the cloudlet in cloud computing is NP-hard problem [20]. This NP hard problem is defined as a platform constituted by potentially large numbers of users from different environments and with different needs and demands on thousands or millions of Virtual Machines waiting for their turn to be served [7,9]. It is the wish of every cloud service provider to ensure that every available resource is fully utilized to avoid resources being idle [10].

Revised Manuscript Received on October 30, 2019.

* Correspondence Author

Keerthika P*, Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode, TamilNadu, India. Email: keerthikame@gmail.com.

Suresh P, Information Technology, Kongu Engineering College, Perundurai, Erode, TamilNadu, India. Email: sureshme@gmail.com

Manjula Devi R, Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode, TamilNadu, India. Email: rmanjuladevi.gem@gmail.com.

Sangeetha M, Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode, TamilNadu, India. Email: sangeethcse@gmail.com.

Sagana C, Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode, TamilNadu, India. Email: sagana.c @kongu.ac.in.

Hence the resource utilization is considered as an important criterion for measuring the performance of a scheduling system. Always task scheduling problem is NP-complete which results in achieving only the near optimal solution and not the exact solution. At present, many heuristic scheduling algorithms are proposed by many researchers. The scheduling algorithms under research falls into four categories: Algorithms based on task duplication, based on task clustering, based on random search and list scheduling [1].

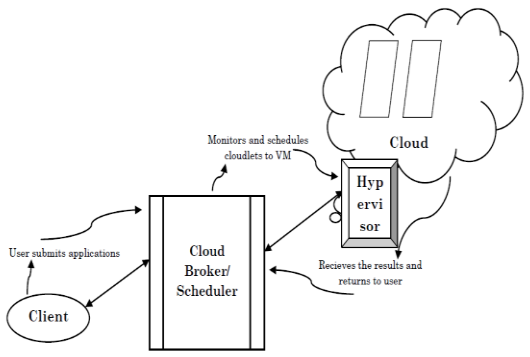
Cloud
Monitors and schedules cloudlets to VM
Hyp ervi sor
User submits applications
Cloud Broker/ Scheduler
Recieves the results and returns to user
Client

Fig.1 Basic Cloud Scheduling Architecture

The basic cloud scheduling architecture is shown in **fig. 1**. The architecture follows a strategy in which the users submit the tasks to the cloud broker. The tasks are split into cloudlets. The cloud broker is also referred as cloud scheduler. The cloud scheduler performs the function of job monitoring and scheduling which helps in selecting the VMs for the submitted cloudlets. The cloud scheduler receives the information about the available VMs from the Hypervisor whish takes care of managing the VMs. An appropriate VM is selected for individual cloudlets. The cloudlets are assigned to the VM and the results are returned back to the Broker. This scheduling process needs to be done in an effective manner in such a way that the completion time of tasks gets minimized. There are many research works proposed by researchers around the world to achieve the efficient and effective scheduling process. The objective of this research is to design a proactive fault tolerant scheduling algorithm that results in minimized completion time and increased user satisfaction.

II. LITERATURE REVIEW

In the current world of cloud computing, there are many research works proposed by several researchers focusing on several issues. In this section, the research works carried out towards scheduling of cloudlets are discussed.

An improved task scheduling algorithm is proposed [1] that works for scientific workflows in cloud environment. The algorithm is based on task duplication and task grouping. Also it focuses on minimizing the total execution time, reducing execution costs of users and improving the resource utilization. A DAG is constructed and converted to tree graph. The tasks are divided into groups so as to reduce the communication overhead. This methodology results in smaller makespan of workflow, fewer processors are used, and has higher resource utilization for both compute-intensive and data-intensive workflow, especially for data-intensive workflow.

An efficient resource provisioning strategies are proposed [2] that focuses on reducing the power consumption while scaling the cloud resources. An auto scaling mechanism has been introduced based on machine learning techniques for time series forecasting and queuing theory. This mechanism concentrates on accurate prediction of processing load of a distributed server and estimation of the appropriate number of resources that must be provisioned. This is done in order to optimize the service response time and fulfill the SLA factors. This algorithm reduces energy consumption and also the infrastructure costs.

A Modified Max-Min (MMax-Min) algorithm is proposed [5], which is an extension of Max-Min algorithm. The MMax-Min algorithm identifies a cloudlet with maximum completion time and a cloudlet with minimum completion time. Based on the requirement of the task such as deadline and cost, the corresponding VM is allocated to the cloudlet selected. The performance of the algorithm is compared with Max-Min and Round Robin algorithms in terms of makespan and load balancing.

An Extended Min-Min algorithm for an effective scheduling [6] that considers the priority scheduling that reduces the overall execution time of all the cloudlets. The performance of the proposed algorithm is compared with other heuristic algorithms such as Min-min, FCFS, Max-Min and Round Robin algorithms. It is inferred that the proposed algorithm results with minimized makespan when compared to other algorithms.

A load balancing intelligent strategy for task scheduling [7] is proposed that makes use of Ant Colony Optimization which allocates tasks to VMs. This algorithm works by considering the load balancing factor, related to the task finishing rate. This makes the task finishing rate at different resource being similar and the ability of the load balancing is improved. A Genetic algorithm based heuristic scheduling for scientific workflows is proposed in [8]. This algorithm acts as an interface between cloud user and provider in receiving, analyzing and distributing the application to the appropriate and available resources. This considers the inter dependencies among the application tasks. This algorithm reduces the makespan without increasing the cost.

A dynamic load balancing algorithm [11] is proposed, that balances the workload among the VM using elastic provisioning and de-provisioning depending upon the last optimal k-interval. This algorithm aims at reducing the makespan and increasing the cloud resource utilization ratio. This algorithm also aims at satisfying the user needs by meeting the QoS parameters

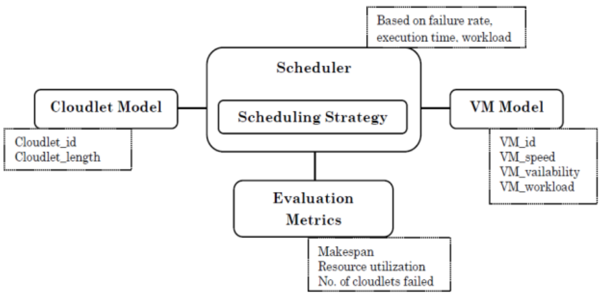

Fig. 2 Cloudlet Scheduling Model with Attributes

defined by the user. A task scheduling algorithm LGSA is proposed in [13] which is the hybrid of lion optimization algorithm. This algorithm tries to resolve the multi-objective optimization scheduling problem. The problems like overloading and under-loading of VMs, cost, energy, resource utilization are addressed.

The traditional Min-Min algorithm is an efficient but simple algorithm that results in minimized makespan than other heuristic algorithms. The biggest drawback of it is unbalanced load of resources. An improved load balanced Min-min algorithm [14] is proposed, that increases the resource utilization. Also the user priority for the tasks is also considered in order to achieve better user satisfaction.

In [17], Scheduling with BAT algorithm is discussed. The BAT algorithm is an optimization algorithm wherein the calculation is a self-learning enhancement calculation. The calculation is done using the idea of biological arrangement of the bats. Bats normally emit ultrasonic pulses on the environment surrounding it. The emitted ultrasonic pulses will be reflected as echoes. The bat catches the echoes produced and with the help of received echoes it identifies the obstacles and preys. This concept can be applied for scheduling problems to find an optimal solution.

A multi-constrained scheduling algorithm [4] is proposed, that considers multiple constraints such as system load, processing cost, processing time, user's deadline and resource failure. A resource allocation algorithm is designed that is budget- constrained and load-constrained with tolerance to failures and also provides user satisfaction. A fault tolerant based Load balancing scheduling system [16] that takes care of both balanced load and failure tolerance is discussed. A proactive fault tolerance mechanism is introduced in which the difference factor is computed based on user deadline and expected completion time. The Resource Selection Parameter is computed which is mainly used for selecting the optimized resource.

A heuristic load balancing algorithm for IaaS clouds is proposed in [15]. An efficient strategy is defined that configures the servers based on the number of tasks submitted and their sizes and identifies the suitable VMs, assigns the tasks to them and tries to maximize the utilization rate of the computing resource.

III. MATERIALS AND METHODS

A. Problem Statement

The fault tolerant scheduling is a problem defined as assigning the cloudlets to the available VMs in a way to minimize: (1) the makespan to execute all the submitted tasks, (2) to improve the resource utilization and (3) to reduce the failure rate of submitted cloudlets. Let the submitted applications be split into number of cloudlets $\{CL_1, CL_2, CL_3, \ldots\ldots CL_n\}$ and the number of VMs available in the cloud provider be $\{VM_1, VM_2, VM_3, \ldots..VM_n\}$. The cloudlets are defined with their characteristics such as cloudlet_Id and Cloudlet_length. The VM are defined with the characteristics such as VM_Id, VM_speed, VM_workload and VM_availability. The Physical Machines (PM) are defined with the number of VMs allocated to it.

B. Cloud Scheduling Model

The cloudlet scheduling model with their characteristics is given below in **fig.2.** This architecture clearly states the characteristics of VM, cloudlet and the performance metrics. The scheduler performs the functionality of finding a correct match for the applications in the form of cloudlets and the resources to execute them in the form of VMs. The scheduler has the ability to completely analyze the characteristics of applications prior to execution and to analyze characteristics of available computing resources and provide the users with the correct matching resources for their applications.

C. Proposed FTVMA Strategy

The proposed FTVMA algorithm follows the scheduling architecture depicted in **fig.3**. User submits the applications $\{A_1, A_2, A_3 \ldots\ldots\ldots\ldots\ldots A_n\}$ to the cloud broker. The cloud broker is responsible for receiving the applications, splitting them into cloudlets $\{CL_1, CL_2, CL_3, \ldots\ldots CL_n\}$. The cloud broker plays the role of VM monitor, Cloudlet Monitor, Failure Detector, Workload Predictor and Scheduler. The VM Monitor is responsible for monitoring the status of the VM in terms of load, availability, number of cloudlets submitted, number of cloudlets successfully completed, number of cloudlets failed to be executed and the list of cloudlets already queued up.

Cloudlet Monitor is responsible for monitoring the status of completion of the cloudlet and its size. The Failure Detector receives the information of number of cloudlets submitted and successfully completed and calculates the failure rate of each VM. The Workload Predictor predicts the current load of the VM by fetching the information from VM Monitor. The Scheduler, as a whole makes use of all these information and identifies an appropriate VM for each cloudlet.

The VMs created in this simulation for scheduling have the properties like speed, failure_rate, ready_time and work_load. The processing capacity or Speed of VM_j can be calculated as

$$S_j = MIPS_j \qquad (1)$$

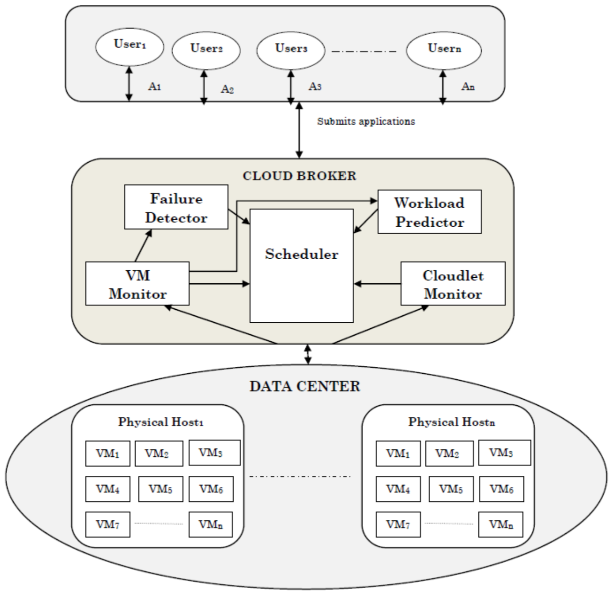

Fig. 3 Proposed FTVMA Scheduling Architecture

The number of cloudlets submitted to VM_j is given by CL_{jsub}. The number of cloudlets successfully completed by the VM_j is given by CL_{jsucc} and the number of cloudlets failed to be executed by VM_j is given by CL_{jf}.

The failure_rate of VM_j can be calculated as

$$FR_j = \frac{CL_{jf}}{CL_{jsub}} \qquad (2)$$

The ready time of VM_j is defined s the time by which the VM completes the previously assigned cloudlets and become ready to execute the current CL_i. This is given by

$$LCT_j = \sum_{k=1}^{p} ETE_{jk} \qquad (3)$$

where k is the number of cloudlets assigned already to VM_j and $ETE_{jk}$is the Expected Time to Execute k cloudlets in VM_j. The Work_load of VM_j is calculated as

$$WLD_j = \sqrt{\sum_{k=1}^{p} a_k {L_k}^2} \qquad (4)$$

where L_k is the load attribute of the VM. In this FTVMA algorithm, the load attribute considered is the CPU utilization in seconds. Henceforth, the load can be given as,

$$WLD_j = \frac{\sum_{j=1}^{n} MI_j}{S_j \times AT_j} \qquad (5)$$

The new_load of the VM_j can be calculated as

$$NLD_j = WLD_j + \frac{MI_i}{S_j \times AT_j} \qquad (6)$$

The Expected Time to Execute can be calculated as,

$$ETE_{i,j} = \frac{Size(CL_i)}{Speed(VM_j)} \qquad (7)$$

The number of cloudlets successfully completed is termed as Hit Count and is defined by

$$HC_{tot} = \sum_{j=1}^{M} HC_j \qquad (8)$$

where HC_j is the hit_count of individual VM which is calculate as

$$HC_j = \frac{CL_{jsucc}}{CL_{jsub}} \qquad (9)$$

The FTVMA algorithm runs in the scheduler of the Cloudsim architecture. The FTVMA Scheduling algorithm is shown in **fig. 4** and the notations used in the algorithm are given in Table 1.

Table 1: Notations and their description

Notations	Description
$A_1, A_2, A_3, \dots\dots\dots A_n$	Applications submitted by the user
$CL_1, CL_2, CL_3, \dots\dots\dots CL_n$	Cloudlets 1 to n
$VM_1, VM_2, VM_3, \dots\dots\dots VM_m$	Virtual Machine in the data center from 1 to m
FR_j	Failure Rate of VM_j
AT_j	Availability time of VM_j
ETE_{jk}	Expected Time to Execute cloudlet k in VM_j
MIPS	Million Instructions per Second
S_j	Speed of VM_j
CL_{jsub}	No. of Cloudlets submitted to VM_j
CL_{jsucc}	No. of Cloudlets successfully completed by VM_j
CL_{jf}	No. of Cloudlets failed to execute in VM_j
LCT_j	Ready Time or Last task Completion Time of VM_j
WLD_j	Workload of VM_j
NLD_j	New Workload of VM_j
$TT_{i,j}$	Total Time to Execute CL_i in VM_j
VM_{util}	Utilization of VM_j
AV_{util}	Average Utilization of all VMs
HC_{tot}	Total Hit count of all VMs.
HC_j	Hit count of VM_j

```
1. Receive the submitted cloudlets $CL_i$ from user area wide Cloudlet Monitor.
2. Collect the available VM's and their information such as $work\_load\ WL_j$, $failure\_rate\ FR_j$, $speed\ S_j$ and $availability\_time\ AT_j$ from VM Monitor.
3. $\forall CL_i \in \{CL_1, CL_2, CL_3, \dots\dots\dots . CL_n\}$
   $\forall VM_j \in \{VM_1, VM_2, VM_3, \dots\dots\dots . VM_m\}$
   do
       Calculate the Expected Time to Execute as follows.
           $ETE_{i,j} = \frac{MI_i}{S_j}$
           where $S_j$ is the processing speed of $VM_j$ in
       million instructions per second and $MI_i$ represents
       the size of $CL_i$ in million instructions.

       Calculate Total Time to Execute $TT_{i,j}$ using the
       equation below.

           $TT_{i,j} = ETE_{i,j} + LCT_j$
   end
4. $\forall CL_i \in \{CL_1, CL_2, CL_3, \dots\dots\dots . CL_n\}$
   do
       Sort the VMs in ascending order based on their $TT_{i,j}$
       Sort the VMs in ascending order based on their $FR_j$
       Select the first VM from the sorted list and assign
         the corresponding cloudlet

       $\forall CL_i$ except allocated $CL_i$,
           Update $TT_{i,j}$ of the selected $VM_j$
           Update $LCT_j$ and $WLD_j$ upon each
               allocation.
   end

5. Calculate Makespan of the schedule using the formula,
       $Makespan = \max\{LCT_j\}$
6. Calculate the Utilization of VM as
   $VM_{util} = \frac{\sum_{j=1}^{n} MI_j}{S_j \times AT_j} \times 100$

7. Calculate Average Utilization given by,
       $AV_{util} = \frac{1}{N} \sum_{j=1}^{N} VM_{util_j}$
8. Calculate the Hit Count as
       $HC_{tot} = \sum_{j=1}^{M} HC_j$
```

Fig. 4 Proposed FTVMA algorithm

IV. RESULTS AND DISCUSSION

The proposed FTVMA algorithm mainly focuses on minimizing the makespan. A proactive fault tolerant technique is introduced that assess the probability of failure at the time of scheduling. The FTVMA is evaluated using the cloudsim simulation environment and the results are compared with the existing scheduling algorithms such as Min-min, LBIMM algorithm [14] and EMin-Min algorithm [6].

A. Experimental Setup

The simulation programs are used for modeling the behavior of the system (CPU, network etc.) by calculating and analyzing the interaction among different entities of the system using mathematical formulas, or by actually capturing the observations from a production system [18]. Cloudsim is a frame work developed by the GRIDS laboratory of university of Melbourne that enables seamless modeling, simulation and experimenting on designing cloud computing infrastructures [18]. The cloudsim simulator environment is used for evaluating the performance of the proposed algorithm. The characteristics and features of cloudsim related to this work is discussed below. The cloudsim architecture shown in **fig. 5** comprises of Data Center, Scheduler wherein the proposed scheduling algorithm is implemented, cloudlets, VMs and the physical host.

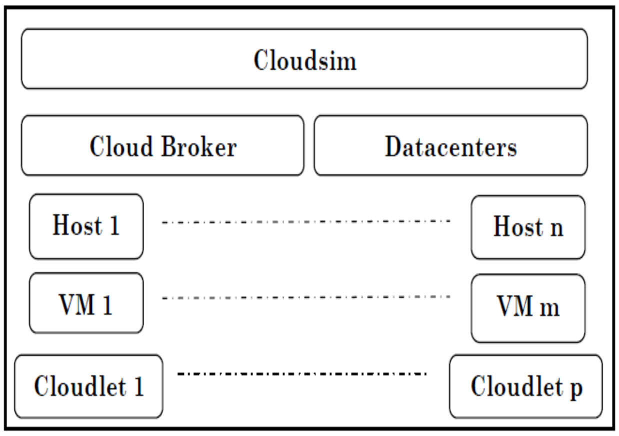

Fig. 5 Cloudsim Architecture

Cloudsim can be used to model data centers, host, service brokers, scheduling and allocation policies of a large scaled cloud platform [19]. Cloud helps in VM provisioning at two levels: At the host level and at the VM level. At the host level, it is possible to specify how much of the overall processing power of each core will be assigned to each VM. At the VM level, the VM assigns a fixed processing power to the individual cloudlets. This is termed as VM Scheduling [19]. Every datacenter should register with Cloud Information Service registry (CIS). The user requests are mapped to suitable cloud providers. The available list of cloud providers are issued by the CIS. Among the list, the cloud providers which are capable of providing the needed services are identified and a perfect matching provider is submitted with the application [18]. The main components of Cloudsim and their relationships are given below.

Host: It models a physical server.

VM: It models a virtual machine which runs on the host.

Cloudlet: It models the cloud-based application services.

Datacenter: It models services offered by cloud providers. It consists of homogeneous or heterogeneous compute hosts.

CIS: An entity that registers datacenter entity and discovers the resource.

VM Allocation: A provisioning policy at the datacenter level. It helps in allocating VMs to hosts.

VM Scheduler: Runs at each host to allocate processors to VMs.

Cloudlet Scheduler: Runs at VM and defines policies to share processing power to the cloudlets in a VM [19].

In this simulation, the number of VMs created is 16 and the number of cloudlets is 512. The characteristics defined for cloudlet are shown in Table 2.

Table 2. Characteristics of Cloudlet

Characteristics	Range
Cloudlet_Id	0 to 511
Cloudlet_Length	2,00,000 MI to 4,00,000 MI
Cloudlet_Size	200 to 500 MB

The characteristics of VMs defined in the simulation are given below in Table 3.

Table 3. Characteristics of VMs

Characteristics	Range
VM_Id	0 to 15
MIPS	50 to 500
VM_ImageSize	1000
No. of CPU for each VM	1
Hypervisor	Xen

B. Performance Metrics

The performance metrics used to evaluate the proposed FTVMA algorithm are makespan, hit_count and the average VM utilization. These metrics clearly defines the performance in such a way that, makespan and hit_count ensures the level of QoS and the average VM utilization ensures the level of QoE. These metrics are defined below.

Makespan: It is defined as the maximum of ready time of all the VMs. The LCT of all the available VMs are calculated and the maximum of all the LCT is termed as makespan and is given by equation (10).

$$Makespan = \max\{LCT_j\} \qquad (10)$$

Hit Count: It is defined as the number of cloudlets successfully completed its execution among the set of cloudlets submitted for that schedule without failure. The hit_count of each VM is calculated using the equation (9) and the overall hit_count is calculated using the equation (8).

Average VM Utilization: The utilization of each VM is calculated with its load assigned to it by the scheduler and is given in the equation (11).

$$VM_{util} = \frac{\sum_{j=1}^{n} MI_j}{S_j \times AT_j} \times 100 \qquad (11)$$

and the Average VM Utilization is given by the equation (12).

$$AV_{util} = \frac{1}{N}\sum_{j=1}^{N} VM_{util_j} \qquad (12)$$

C. Experimental Results

The proposed FTVMA algorithm is evaluated in the above mentioned experimental setup and the results obtained are compared with the existing algorithms Min-min, LBIMM algorithm [14] and EMin-Min algorithm [6].

The data sets taken for evaluation are categorized into 4 different cases of 512 cloudlets and 16 VMs each.

- ***Case 1*** depicts the data with less performing VMs and cloudlets that requires low performing resources for execution (LVLC).
- ***Case 2*** depicts the data with highly performing VMs and huge cloudlets that requires high performing resources for execution (HVHC).
- ***Case 3*** depicts the data with less performing VMs and huge cloudlets that requires high performing resources for execution (LVHC).
- ***Case 4*** depicts the data with highly performing VMs and cloudlets that requires low performing resources for execution (HVLC).

Since the different combinations taken represent all types of real time scenarios, the data are grouped in these four cases. The performance analysis of FTVMA based on makespan is depicted in fig. 6. The results shows that the proposed FTVMA has comparatively reduced makespan than the existing algorithms Min-min, EMin-min and LBIMM. Since FTVMA allocates VMs based on failure rate and minimum execution time, the makespan is minimized.

Hence the proposed algorithm has relatively minimized makespan than the other algorithms. In case 4, the data considered is the high performing VM and less requirement cloudlet. Hence there is a significant reduce in makespan of all the algorithms.

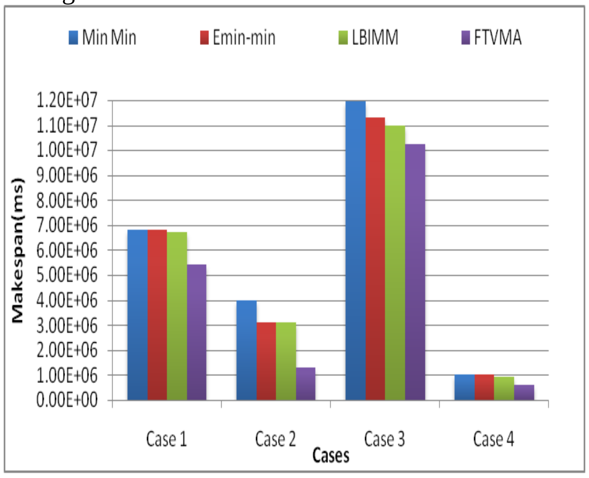

Fig. 6 Performance Analysis of FTVMA based on makespan in milliseconds

An important metric to measure the performance based on failure rate is the hit_count. Based on hit_count, when compared, the proposed FTVMA shows higher number of successfully completed cloutlets without any failure. The comparison is depicted in fig.7 and it shows that the LBIMM has reduced hit_count than the other algorithms since it doesn't concentrate on failure rate of VMs. The proposed FTVMA algorithm has increased hit_count when compared to other algorithms.

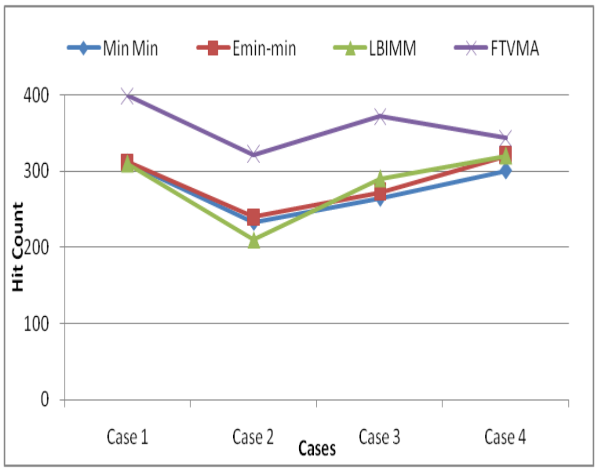

Fig. 7 Performance Analysis of FTVMA based on hit_count

In terms of VM utilization, the metric used to measure the performance is the average VM utilization. The comparison is shown in fig.8, from which it is inferred that the proposed FTVMA algorithm has increased VM utilization on an average of 89% for all four cases. Irrespective of cases, the VM utilization seems to have similar range of values in percentage.

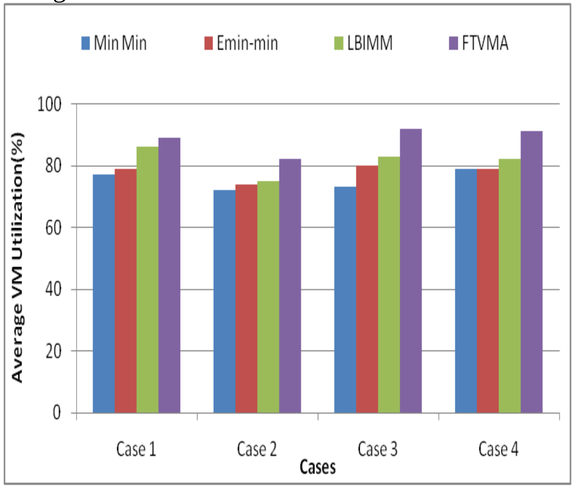

Fig. 8 Performance Analysis of FTVMA based on Average VM Utilization

V. CONCLUSION AND FUTURE WORK

In this work, a fault tolerant based scheduling algorithm is proposed that positively finds a perfect match of VM considering the failure rate of VMs and also in view of reducing the makespan. The proposed algorithm is tested in different scenarios where in the low requirement cloudlets submitted to high performing VMs, high requirement cloudlets submitted to less performing VMs, the low requirement cloudlets submitted to low performing VMs and the high requirement cloudlets submitted to high performing VMs. In our real time cloud environment these four categories are necessarily to be addressed. The proposed FTVMA algorithm works by considering all these factors and results in reduced makespan, improved hit_count and VM utilization. This work can be extended in terms of considering other measures such as scalability by considering the overloaded and underloaded VMs. This can be further extended with dynamic VM consolidation methodologies.

REFERENCES

1. X. Geng, Y. Mao, M. Xiong, Y. Liu, (2018), "An improved task scheduling algorithm for scientific workflow in cloud computing environment", Cluster Computing., pp. 1-10. DOI https://doi.org/10.1007/s10586-018-1856-1
2. Rafael Moreno-Vozmediano., Rubén S. Montero., Eduardo Huedo and Ignacio M. Llorente, (2019), "Efficient resource provisioning for elastic Cloud services based on machine learning techniques", Journal of Cloud Computing: Advances, Systems and Application. DOI: https://doi.org/10.1186/s13677-019-0128-9
3. Karthikeyan, T., Vinothkumar, A., Ramasamy, P.,(2019), "Priority Based Scheduling in Cloud Computing Based on Task—Aware Technique", Journal of Computational and Theoretical Nanoscience, Volume 16, Numbers 5-6, pp. 1942-1946(5). DOI: https://doi.org/10.1166/jctn.2019.7828
4. P.Keerthika and P.Suresh, (2015), "A Multiconstrained Grid Scheduling Algorithm with Load Balancing and Fault Tolerance", The Scientific World Journal, Vol.2015, Article ID 349576, 10 pages.
5. Konjaang, J., Ayob, F. H., & Muhammed, A. (2017), "An optimized Max-Min scheduling algorithm in cloud computing", Journal of Theoretical & Applied Information Technology, 95(9), 1916-1926.
6. J. Y. Maipan-uku., A. Mishra., A. Abdulganiyu., A. Abdulkadir.,(2018), "An Extended Min-min scheduling algorithm in cloud computing", i-manager's Journal on Cloud Computing,Vol. 5,No. 2.
7. Keshk, A. E., El-Sisi, A. B., & Tawfeek, M. A. (2014). "Cloud task scheduling for load balancing based on intelligent strategy", International Journal of Intelligent Systems and Applications, 6(5), 25-36.
8. Israel Casas, Javid Taheric, Rajiv Ranjan, LizheWang, Albert Y .Zomaya, (2018), "GA-ETI:An enhanced genetic algorithm for the scheduling of scientific workflows in cloud environments", Journal of Computational Science, Vol 26, pp.318-331.
9. Huang, Q. Y., & Huang, T. L. (2010), "An optimistic job scheduling strategy based on QoS for Cloud Computing", In Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on (pp. 673-675). IEEE.
10. Choudhary, M., & Peddoju, S. K. (2012). "A dynamic optimization algorithm for task scheduling in cloud environment", International Journal of Engineering Research and Applications (IJERA), 2(3), 2564-2568.
11. M. Kumar, S. Sharma,(2018), "Deadline constrained based dynamic load balancing algorithm with elasticity in cloud environment", Computers & Electrical Engineering, vol. 69, pp. 395-411.
12. Mell, P., Grance, T., (2011). "The NIST Definition of Cloud Computing", vol. 145. NIST Spec. Publ., p. 7
13. N. Manikandan.A. Pravin., (2019), "LGSA: Hybrid Task Scheduling in Multi ObjectiveFunctionality in Cloud Computing Environment", 3D Res, pp.1 -16. DOI: https://doi.org/10.1007/s13319-019-0222-2
14. H. Chen, F. Wang, N. Helian, G. Akanmu, (2013), "User-priority guided min-min scheduling algorithm for load balancing in cloud computing", 2013 Natl.Conf. Parallel Comput. Technol. Parcomptech.
15. M. Adhikari, T. Amgoth, (2018), "Heuristic-based load-balancing algorithm for IaaS cloud", Future Generation Computer Systems, vol. 81, pp. 156-165.
16. P.Suresh and P.Keerthika, (2016), "Design of a Fault Tolerant Load Balancing Scheduler for Computational Grid", Asian Journal of Research in Social Sciences and Humanities Vol. 6, No. 5, May 2016, pp. 762-774.
17. Praveen V, Keerthika P, Saran Kumar A, Siva Priya G.,(2019), "A Survey on Various Optimization Algorithms to Solve Vehicle Routing Problem", 5th International Conference on Advanced Computing & Communication Systems (ICACCS).
18. Buyya, R., Ranjan, R., Calheiros, R.N.,(2009), "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities", in Proceedings of the 7th High Performance Computing and Simulation (HPCS 2009) Conference, Leipzig, Germany.
19. Ghalem, B., Fatima Zohra, T., and Wieme, Z. (2010), "Approaches to Improve the Resources Management in the Simulator CloudSim" in ICICA 2010, LNCS 6377, pp. 189–196.
20. Mohammad Masdari, Sima ValiKardan, Zahra Shahi, Sonay Imani Azar (2016), "Towards workflow scheduling in cloud computing", Journal of Network and Computer Applications, Volume 66 Issue C, May 2016, Pages 64-82.

AUTHORS PROFILE

Dr.P.Keerthika is currently working as Senior Assistant Professor in the Department of Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode. She has a work experience of 13 years in teaching. She has completed her Ph.D in Anna University, Chennai. She is contributing towards research in the area of Grid and Cloud Computing for the past 9 years. She has published around 20 research papers in reputed journals indexed by Scopus and SCI and presented 15 papers in National and International Conferences. Her areas of interest are Machine Learning, Cloud Computing and IoT. She is one of the recognized Supervisors in the Faculty of Information and Communication Engineering under Anna University, Chennai. She is currently guiding Ph.D scholars in the areas of Machine Learning and Cloud. She has received grants from various funding agencies like DRDO and DST. She is currently acting as Reviewer and Editorial Board member in reputed journals.

Dr.P.Suresh is currently working as Senior Assistant Professor in the Department of Information Technology, Kongu Engineering College, Perundurai, Erode. He has a work experience of 13 years in teaching. He has completed his Ph.D in Anna University, Chennai. He is contributing towards research in the area of Grid and Cloud Computing for the past 9 years. He has published 24 research papers in reputed journals indexed by Scopus and SCI and presented 25 papers in National and International Conferences. His areas of interest are Networks, Cloud Computing and Big data. Currently he is working in the areas of Machine Learning and IoT. He is one of the recognized Supervisors in the Faculty of Information and Communication Engineering under Anna University, Chennai. He is currently guiding Ph.D scholars in the areas of Machine Learning and Cloud. He has received grants from various funding agencies like DRDO, ICMR, and CSIR towards the conduct of seminars and workshops. He has received Best Faculty Award from Kongu Engineering College. He is currently acting as Reviewer and Editorial Board member in reputed journals.

Dr. R. Manjula Devi is a senior Assistant Professor in the Department of Computer Science and Engineering at Kongu Engineering College, Perundurai. . She has a work experience of 13 years in teaching. She completed her PhD (Information and Communication Engineering) from Anna University, Chennai (2015). Her research interests are Machine Learning, Soft Computing, Computer Graphics, and Artificial Intelligence. She has published over 25 technical papers on Neural Network, Intrusion Detection System, Data Mining and Cloud Computing. She is currently acting as Reviewer and Editorial Board member in reputed journals. She is authored more than 15 books edited by Charulatha Publications, Chennai. She has been honored with the various award such as Best Faculty Award, Shri P. K. Das Memorial Best Faculty Award & Life Time Achievement Award–2015, Best Author Award, GRABS Best Young Teacher Award, Young Woman Educator & Scholar Award, etc.,

Ms.M.Sangeetha is currently working as Assistant Professor in the Department of Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode. She has a work experience of 9 years in teaching. She has pursuring her Ph.D in Anna University, Chennai. She is contributing towards research in the area of Machine Learning for the past 9 years. She has published around 3 research papers in reputed journals indexed by Scopus and presented 7 papers in National and International Conferences. Her areas of interest are Machine Learning and Graph theory.

Ms. C.Sagana is currently working as Assistant Professor in the Department of Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode. She has a work experience of 6 years in teaching. She has completed her M.E in Kongu Engineering College. She is contributing towards research in the area of Cloud Computing for the past 5 years. She has published 2 esearch papers in reputed journals indexed by Scopus and presented 5 papers in National and International Conferences. Her areas of interest are Machine Learning, Cloud Computing and Web Technology.