# A Reactive Fault Tolerance Approach For Cloud Computing

Eman AbdElfattah, Mohamed Elkawkagy and Ashraf El-Sisi
Computer Science Dept., Faculty of Computers and Information
Menoufia University,
Egypt
eman4cs@gmail.com, M_nabil_shams@yahoo.com, ashraf.elsisi@ci.menofia.edu.eg

***Abstract*— Reliability is a critical requirement for any system. To achieve high reliability the fault tolerance must be accomplished. Fault tolerance refers to the task must be executed even in occurring the fault. Cloud computing has emerged that grants users with access to remote computing resources. Although the current development of the cloud computing technology there are more challenges and chances of errors occur during execution. In this paper, the proposed model tolerates the faults by using replication and resubmission techniques. Then it decides which the best virtual machine depending on the reliability assessments. Then it reschedules the task once the failure occurs to the highest reliability processing node instead of replicating this task to all available nodes. Additionally, we compare our proposed model with another model that used replication and resubmission without any improvement. And we evaluate the experiments by a CloudSim simulator. We conclude that the proposed model can provide comparable performance with the traditional replication and resubmission techniques.**

***Keywords*— *Cloud computing; fault tolerance; virtual machine; replication; resubmission; cloudsim.***

## I. INTRODUCTION

Recently cloud computing is considered as a hot topic that grants the client to use any resources needed at any time. Many users compute a lot of tasks at the same time on cloud service. So, it should be more stable and highly reliable to satisfy the requirements of users. As show at Fig.1 a cloud computing environment is classified to 3 layers:

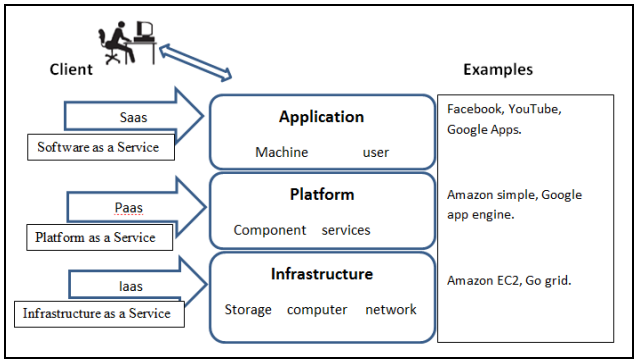


Fig.1. Cloud computing layers.

Application, platform and infrastructure. All of these layers will describe below: The application layer is at the highest level of the hierarchy which consists of the actual cloud application where the client sends the application. The platform layer built on below the application layer which consists of operating system and application framework .Also used to minimize the load of applications on vms. The infrastructure layer or virtualization layer which contains the cloud components. Cloud services can be classified to 3 categories:

IaaS refers to infrastructure as a service which contains the resources that users need, its provider are contain amazon EC2, go grid. PaaS refers to platform as a service, its provider include Google app engine, amazon simple. SaaS refers to software as a service, for example of its provider are Google apps, Facebook application.

There are also three types of failure: hardware, vm and application failure. But there is more than one fault such as a permanent fault that means there is a mistake in the component. A transient fault means fault takes some time until all component functionality restored [11].

Any fault occurs during the execution leads to re-running the application from beginning in the same machine. Alternatively, add a new machine that will cost time, money and resources. Therefore, we need to minimize failure impact on the system and application execution. The failures should be handled by using Fault tolerance techniques. Fault tolerance ensures more availability and reliability of services (VM) during application execution. Reliability of system is referred to the ability of the machine to complete the execution of application successfully understated condition. To handle the fault that may occur there are different types of techniques of fault tolerance that classified in two categories:

The first one is a proactive Fault Tolerance that foresees the problems before it comes and avoids its Influence on the task. Also, it prevents the tasks and VMS from fail and makes sure the task gets done correctly. Some a proactive fault tolerance strategies are as follows [9] [7]. Preemptive migration: that used a control system where tasks are surveillance. When predicting a task will fail to complete its execution on one VM, the task will shift to another virtual machine. Its disadvantage is wasting time for calling another machine [12] [14]. Time check: it predicts the time which any task needs to complete its

978-1-5386-4266-5/17/$31.00 ©2017 IEEE

job by using watch dog. If task not completed in this time, the action is taken [12] [17].

And the second category is a reactive Fault Tolerance Once the fault done in computing node or task it begins to reduce its efforts as far as possible to ensure the task completed its job. Some a reactive fault tolerance strategies are as follows [11] [18]. Replication: This technique used to replicate the task on more than one resource. This repetition increases the chance of at least one task completes its job correctly. Also, it increases the reliability of cloud computing and its availability. Its foible is waste resources utilization because of redundancy [16] [17]. Task resubmission: when the task fails it will re-running again in either the same node or to a different resource for execution from the beginning of the task. It increases the execution time over the expected time because of the repetition of the task [10] [11]. On the other side, there is another technique called Retry: That is the simplest technique and different from the resubmission technique. The retry technique re-execute the failed task in the same resource, but it consumes the execution time like the resubmission technique [5] [14]. Check pointing/Restart: It takes snapshots at different times during task execution to re-running the task when failed from the last point, not from the beginning. However, it requires large storage device when to store its snapshot [15] [16].

On our proposed model, we use replication and resubmission techniques. The traditional replication technique consumes resources but guarantees probabilities of task execution [1]. On the other side, the traditional resubmission technique increases makespan tasks. Makespan is defined as the difference time between start and finish of a sequence of tasks [3]. But the replication and resubmission together prevent consuming resources and time by scheduling the tasks and reallocating the failed task to the highest reliability processing node instead of allocating it to all available nodes. The scheduling point [11] is important when there are many tasks need to execute at the same time. It starts to distribute the tasks to available vms to executing. Scheduling system should be fault tolerant for the failures that occur in cloud computing environment [3].

The rest of the paper has the following sections: section II discusses the previous work, section III describes the proposed model mechanism, We show the experimental results in section IV. Conclusion and future work will be discussed at section V.

## II. PREVIOS WORK

There are many research points at fault tolerance field such as: author [2] Focus on calculating the reliability of the virtual machine. The researcher is interested in reliability assessment and used replication technique to replicate the task to all available vms who rent from cloud computing. Also, calculate time and define task status. Depending on reliability assessment the system decides which VM is the best. The best VM it has a high-reliability assessment. If the task fails in all VM, the systems will backward recovery to this task and rerunning again in a new machine. This technique needs a large storage device and more available nodes to be added when failure continue to occur and waste in resource utilization because of backward recovery. On the other hand, author [1] presented another technique depends on replication and resubmission technique to tolerate the faults. Start when client send a task to cloud computing. the preprocessing module begins to prepare the task and send it to replication based scheduling module which repeats the task to all available vms. The executor module starts to run the task. If any copy of task passed and complete correctly, the system starts the next task. But if all VM failed to execute the task, then the tasks are resubmitted again to the same node with a new set of parameters. If the workflow continues in failing and reached to the failure factor, it is considered a new task and rescheduled on a different node. In case of failing again, the processor sends rejected message to the user. This research needs more resources and waste time because of rerunning of tasks more than one time. After that, author [4] introduced the model to design a fault tolerance scheduling algorithm to satisfy the reliability requirement with minimum resources. This followed by presenting a high reliability and low redundancy storage architecture for cloud computing [5]. Author [18] used migration algorithm to balances the load on the host to preserves the level of fault tolerance depending on load balancing techniques. Author [19] build model depends on CPU temperature to predict a problem on the physical machine and using migration algorithm to migrate VMs to some optimal physical machine.

## III. PROPOSED MODEL MECHANISM

The proposed model can schedule tasks in the presence of failure by using cloud simulator using (FCFS) schedule algorithm. Failure is generated randomly by generate an injection error randomly at tasks (T3, T5, T8). Before describe our proposed model first we need to describe two algorithms: The first one is that used replication and resubmission without any improvement. It starts take the tasks from the user and replicate every task to all available virtual machines. If at least one of these tasks passed then send a message to DCB to start another one task. But if the task that replicated failed it starts to reschedule this task again to all available VMS. Then resubmit it to start execution again from the beginning. This algorithm wastes in resources because of replication technique. And spend more time because of resubmission technique. And if also failed again this algorithm repeating this process until reach to the number of resubmission factors that defended. When the resubmission factor increased over times defended, the DCB sends a message to the user with rejected this task.

The second algorithm which is the proposed model the client begins to submit tasks to cloud computing. Then the Data Center Broker (DCB) initiates the independent queue (indtask queue) to Store tasks. Then the information of tasks is stored in the scheduled queue. The DCB begin to distribute the task to all available VMs and start to execute the task. When the task status is passed the scheduled queue update its information to send the next ready task to DCB to execute. If the task fails to complete its execution in all available VMs, the DCB re-schedule this task to the highest reliability. After each cycle, the reliability is increased when status is passed and decreased when status is failed. To reduce the resource consumption and the makespan, our technique chooses the best VM that is the high reliability assessment to re-schedule the failed task instead of replicating the failed task to all available VMs again. This means that the tasks will be replicated by

resubmission factor if the task fails more than resubmission factor. Moreover, then it will be rejected by DCB through sending a message to the user with the rejected task.

### *A. Proposed Model Steps*

The proposed model deals with faults that occur in all available virtual machine. Where it tolerates the faults basis on reliability assessment and reschedules this task to the best virtual machine reliability. Initially, we have N available node used to batch (replicate) each task to these nodes. Moreover, then calculate the reliability of each node after each cycle. Note that this calculation depends on some factor such as adaptability factor (n), Reliability Factor (RF), maximum reliability (maxrel), minimum reliability (minrel), resubmission factor (resfactor). Now we will explain the steps of the proposed model. The functionality of our model can be divided into a set of events as shown in Fig.2. The set of events are as following: Event 1: tasks submission (line 1-3),When the user submits the application to cloud computing, the data center broker (DCB) begins to initiate some data structure such as an "independent task queue" to store tasks. The information of tasks is stored in the "schedule queue." Event 2: tasks distribution (line 4), At this time, the DCB replicates the ready task to the available rent node by using replication technique.

**Input:** Tasks
**Output:**tasks status, Reliability assessment, Execution time

1. Start
2. initiate Indtask queue , scheduled queue.
3. Initiate reliability, n, RF, minrel, maxrel, resfactor.
4. Distribute tasks to available node
5. if node status = pass then
6. reliability = reliability + (reliability*RF)
7. if (n>1) then n=n-1
8. else if (node status = fail ) then
9. reliability=reliability-(reliability*RF*n)
10. resfactor increase by 1 , n=n+1
11. if reliability >=maxrel
12. reliability=maxrel
13. if reliability<minrel
14. node status=stoped
15. the best reliability= the highest reliability node
16. if (all VM fail to execute the task )then
17. if (resfactor <=3) then
18. Rescheduled the fail task to the best node
19. Recalculate the reliability
20. Else
21. Find the next high-reliability node to resubmit the task
22. If (still fail) then
23. Send Reject messageto(DCB)
24. End

Fig.2. Proposed Model Steps

Event 3: task success (line 5-7), When the task completes its execution successfully, it sends a notification to the global scheduler to start the ready new task. Event 4: VM failure and less than resubmission factor (line 8-19) when a task fails to complete its execution in all available VMs, the DCB reschedules the task to the highest VM reliability. Event 5: VM failure and up to resubmission factor (line 20-24) find the next highest VM reliability to resubmit the task again if more failure the DCB will reject this task and tell the user.

## IV. EXPERIMENTAL RESULTS

This section describes two models. The first model shows the replication and resubmission without any improvements. The second is our proposed model which using the first model with reliability assessment for each node (VM). The main factor of our proposed model is reliability assessment so consider the initial reliability of each VM is set to 1. An adaptability factor n is control in reliability calculation in each VM according to task status. Another factor is taken in our mind Reliability Factor (RF) which is control the increasing or decreasing value of the reliability. Minimum Reliability is the lower level of reliability. The VM cannot execute any task in case of the value of reliability reached to the minimum value. In contrast, the maximum Reliability is the highest level of reliability. The value of reliability must not exceed the value of the maximum value of reliability. Rest of this section will discuss the cloudsim toolkit and experimental results based proposed model.

### *A. Cloudsim Toolkit*

We have used cloud simulator (cloudsim) in our proposed model[22]. Cloudsim is a new framework that allows for modeling and simulation of cloud computing infrastructure and service. Fig.3 shows cloudsim entities. Once user sends tasks to cloudsim, the data center broker begins to schedule these tasks to available VMS and start execution. In this paper, we focus on our work between users, tasks and virtual machines scheduling. The parameters setting of cloud simulator we used in our experiments shown at Table I. we test from 10 tasks to 100 with random size begin with 1000 byte. And increase number of virtual machine from 3 to 10.

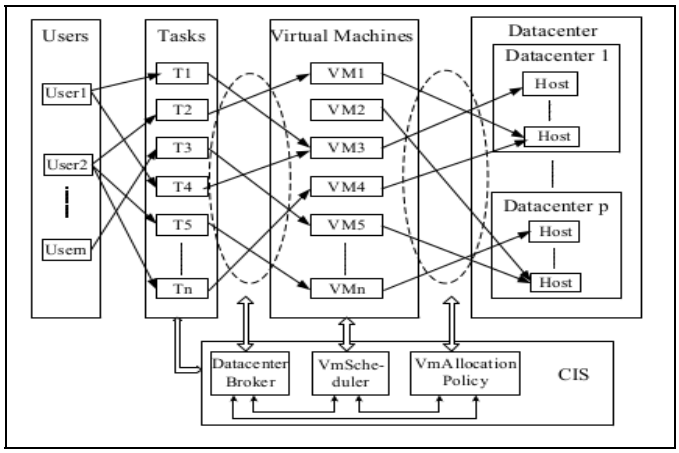


Fig.3. Style of cloudsim entities.

Table I. Parameters setting of cloud simulator

| | |
|---|---|
| **Number of tasks** | 10:100 |
| **Length of one task** | Random start with 1000 byte |
| **Number of VM** | 3:10 |
| **Ram** | 1024-20124 |
| **Mips** | 500 |
| **Number of datacenters** | 2 |
| **Number of hosts** | 2 |

### *B. Replication and Resubmission technique*

Our work depends on a reactive fault tolerance techniques replication and resubmission integrated with reliability assessment for more improvement. Comparison between the traditional replication and resubmission model with our proposed model is considered. Table III, Table IV shows the results of using 3 VM and ten tasks. Each task needs one cycle to execute. In the one cycle, at the same time, we dispatch the task to all VMs (3 VMs). Every cycle begins at a different time, while all VMs begin at the same time. From our experiments, we found this initial variable give the best results as follows table II:

TABLE II. Values for proposed model.

| *Variable* | *Reliability* | *RF* | *Maxrel* | *Minrel* | *resfactor* |
|---|---|---|---|---|---|
| ***Value*** | 1 | 0.02 | 1.2 | 0.7 | 3 |

We defined an independent task queue to receive the recent task from the user. After dispatching operation, every task is sent to all available VMs by replication techniques. Every virtual machine starts the task execution and sends its result to DCB to decide which the best VM. Then begin to reschedule if failure done in all virtual Machine. As shown in table III, and table IV Status (time) means the task is passed with time (t) or means the task is failed with time (t). Scenarios of traditional replication and resubmission model and our improvement model explained as following.

Scenario-1 Replication and Resubmission model

Table III describes the results of replication and Resubmission technique without any improvement (traditional model) which used to tolerate any faults until the task completes its execution. At cycle one the task passed in all VMS but with different time. However, in cycles 2 and 3, the task is passed at least one VM. At cycle 4 task failed on all vms so fault tolerance algorithm begins to tolerate the task to execute during error detection in all VMs. In this case, the second round (2nd cycle) in cycle 4 means that the fault tolerance begins to tolerate this fault by resubmitting the failed task to all available VMs with time (T) again. Whereas the execution time, in this cycle, equals a total execution time of the first and second round of the same cycle 4. The fault also done at cycle 6,9 as shown at table III where the execution time at the second round of cycle 9 is more than 14 milliseconds.

Scenario-2 Proposed model

Table IV describe the results of our proposed model. At cycle one the task passed in all VMS but with different time. Also, we calculate the best VM which is VM1. This means the reliability value of VM1 is the highest reliability. However, in cycles 2 and 3, the task is passed at least one VM, and the best VM is VM2. at cycle 4 shows how fault tolerance algorithm can tolerate the task to execute during error detection in all VMs. In this case, at second round we resubmit the task to the best VM with time (T) Instead of all available vms. Whereas the execution time, in this cycle, equals a total execution time of the first and second round of the same cycle 4. The fault also done at cycle 6,9 as shown at table VI where the execution time at the second round of cycle 9 is more than 3 milliseconds.

After comparing scenario-1 with scenario-2 we conclude that the proposed model can decrease the resource Usability and time execution. At scenario-1 uses all available VMs and execution time is more than 14 milliseconds. The proposed model uses only the best reliability VM and the execution time is less than 4 milliseconds.

TABLE III. Replication and Resubmission Model

| *Task* | *VM1* | *VM2* | *VM3* |
|---|---|---|---|
| | ***Status(time in ms)*** | ***Status(time in ms)*** | ***Status(time in ms)*** |
| 1 | Pass ( 0.563) | Pass (0.749) | Pass (0.857) |
| 2 | Fail (0.208) | Pass (0.321) | Pass (0.423) |
| 3 | Pass (0.151) | Fail ( 0.263) | Fail (0.371) |
| 4 | Fail (0.160) | Fail (0.272) | Fail (0.411) |
| | 2nd cycle | | |
| FT | Pass (1.068 ) | Pass (1.425) | Pass(1.742) |
| 5 | Fail (0.266) | Fail (0.384) | Pass (0.495) |
| 6 | Fail (0.174) | Fail (0.278) | Fail (0.649) |
| | 2nd cycle | | |
| FT | Fail (4.931) | Pass (5.207) | Pass (6.691) |
| 7 | Pass (0.466) | Fail (0.687) | Pass (0.909) |
| 8 | Fail (0.321) | Pass (0.552) | Pass (0.767) |
| 9 | Fail (0.324) | Fail (3.382) | Fail (3.716) |
| | 2nd cycle | | |
| FT | Fail (1.15) | Pass (13.77) | Pass(14.38) |
| 10 | Fail (.468) | Fail (.664) | Pass (0.833) |

TABLE IV. Proposed Model

| *Task* | *VM1* | *VM2* | *VM3* | *Highest Reliability* |
|---|---|---|---|---|
| | ***Status(time in ms)*** | ***Status(time in ms)*** | ***Status(time in ms)*** | |
| 1 | Pass (0.429) | Pass (0.664) | Pass (0.075) | VM1 (1.02) |
| 2 | Fail (0.199) | Pass (0.278) | Pass (0.358) | VM2(1.0404)) |
| 3 | Pass (0.141) | Fail (0.219) | Fail (0.299) | VM2 (1.0196) |
| 4 | Fail (0.149) | Fail (0.249) | Fail (0.326) | VM2 (0.999) |
| | 2nd cycle | | | |
| FT | | Pass (0.616) | | VM2 (1.0192) |
| 5 | Fail (0.168) | Fail (0.263) | Pass (0.342) | VM2 (.999) |
| 6 | Fail (0.150) | Fail (0.239) | Fail (0.329) | VM2 (.979) |
| | 2nd cycle | | | |
| FT | | Pass(0.546) | | VM2 (.999) |
| 7 | Pass (0.151) | Fail (0.241) | Pass (0.325) | VM2 (1.019) |
| 8 | Fail (0.143) | Pass (0.235) | Pass (0.319) | VM2 (1.039) |
| 9 | Fail (3.013) | Fail (3.123) | Fail (3.218) | VM2 (1.019) |
| | 2nd cycle | | | |
| FT | | Pass (3.56) | | VM2 (1.039) |
| 10 | Fail (0.158) | Fail (0.300) | Pass (0.394) | VM2 (1.019) |

## V. CONCLUSION AND FUTURE WORK

In this work proposed fault tolerance technique, allows users to execute their tasks in-spite of different failures that occur in the environment according to the highest reliability of the available VM. In our Experiments, we conducted to test the proposed technique with random generation of tasks in a simulated cloud computing environment (CloudSim) with

simulated faults. The initial implementation of this technique runs over three VMs. The results of these experiments were compared with replication and resubmission technique. This showed that the proposed algorithm able reduce the wastage of resources by rescheduling the failed task to one VM that is the highest reliability assessment. In future, we will work to increase the utilization of resources more than we achieved and used workflow task instead of independent tasks.

## REFERENCES


[1] Jayadivya S K et al.," Fault-tolerant workflow scheduling based on replication and resubmission of tasks in Cloud Computing," International Journal of Computer Science and Engineering, 06 June 2012.

[2] Sheheryar Malik, Fabrice Huet," Adaptive fault tolerance in real-time cloud computing," IEEE World Congress on Services, 2011.

[3] Jing Mei • Kenli Li • Xu Zhou • Keqin Li, "Fault-tolerant dynamic rescheduling for heterogeneous computing systems" Grid Computing,2015

[4] Laiping Zhao, Kouichi Sakurai "A reliability analysis based scheduling algorithm in the heterogeneous system" IPSJ SIG Technical Report, vol 2010.

[5] Qingqing Feng, Jizhong Han, Yun gae, and Dan meng " Magicube:" High reliability and low redundancy storage architecture for cloud computing", IEEE Seventh International Conference on Networking, Architecture, and Storage, 2012.

[6] Anjali D.Meshram, A.S.Sambare, S.D.Zade "Fault tolerance model for reliable cloud computing" International Journal on Recent and Innovation Trends in Computing and Communication, 2013.

[7] ChaonanWang, et al.," Processing time analysis of cloud services with re-trying fault tolerance technique," First IEEE International Conference on Communications in China,2012.

[8] Mohammed el Mehdi Diouri, Olivier Gl¨uck, Laurent Lefevre" A Framework to estimate energy consumption of fault tolerance protocols for hpc applications" 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, 2013.

[9] Harpreet Kaur, Amritpal AL Kaur,"Asurvey on fault tolerance techniques in cloud computing", International Journal of Science, Engineering and Technology,2015.

[10] Anju Bala, InderveerChana," Fault tolerance- challenges, techniques, and implementation in cloud computing" IJCSI International Journal of Computer Science Issues,2012.

[11] P.LatchoumyandP.Sheik Abdul Khader,"Survey on fault tolerance in grid computing", IJCSI International Journal of Computer Science Issues, 2011.

[12] Amal Ganesh, M.Sandhya, Sharmila Shankar" A study on fault tolerance methods in cloud computing,"2014 IEEE.

[13] Daeyong Jung1, SungHo Chin2, Kwang Sik Chung3, and HeonChang Yu1, " Vm migration for fault tolerance in spot instance based cloud computing" , J.J. Park et al. (Eds), 2013.

[14] Qi Zhang • Lu Cheng • Raouf Boutaba," Cloud computing: state-of-the-art and research challenges", J Internet Serv Appl, 2010.

[15] HARPREET KAUR, AMRITPAL KAUR, "A survey on fault tolerance techniques in cloud computing", International Journal of Science, Engineering, and Technology, 2015.

[16] SunilGavaskar.P, Subbarao Ch D.V," A survey of distributed fault tolerance strategies", International Journal of Advanced Research in Computer and Communication Engineering, 2013.

[17] Virendra Singh Kushwah, Sandip Kumar Goyal and Priusha Narwariya " A survey on various fault tolerant approaches for cloud environment during load balancing,", international journal of computer networking, wireless and mobile communications(IJCNWMC),2014.

[18] Amal Ganesh,Dr.M.Sandhya,Dr.Sharmila Shankar"A study on fault tolerance methods in cloud computing"ieee,2014.

[19] G.Gayathri, R.Latha" Implementing a fault tolerance enabled load balancing algorithm in the cloud computing environment," International journal of engineering development and research,2017.

[20] Jialei Liu, Shangguang Wang" Using proactive fault-tolerance approach to enhance cloud service reliability," IEEE transaction on cloud computing,2016.

[21] Alain Techana, Laurent Broto, Daniel Hagimont, "Fault tolerance approaches in cloud computing infrastructures", The eight international conference on autonomic and autonomous system,2012.

[22] Saurabh Kumar Garg and Rajkumar Buyya,"Network cloudsim: modelling parallel applications in cloud simulations", Fourth IEEE International Conference on Utility and Cloud Computing,2011.

[23] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose and Rajkumar Buyya1,"CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms",2010