# Reliability Framework for Cloud Virtual Nodes using Fault Tolerance Techniques

**Mridula Dhingra, Neha Gupta**

***Abstract—*** ***Since cloud computing has numerous benefits and demands, it is much efficient performing real time computing in cloud infrastructure. This real time system takes benefits of scalable virtualized environment and intensive cloud computing capabilities for executing real time tasks. Most of the real time applications undergone its processing stage on remote cloud computing nodes. Here chances of error occurrence is also higher due to undecided latency. And it is necessary that the safety critical applications should have higher reliability. To determine whether the fault tolerance mechanism is having the higher reliability and availability or not. This paper proposes an efficient adaptive fault tolerance mechanism through reliability assessment architecture to enhance the reliability of the system. The decision mechanism is also proposed in this scheme to improve the efficiency a bit higher than normal scheme. At last the fine grained check point algorithm is utilized for reducing the latency. Thus from the analysis of the data it is proved that the execution time of proposed method hikes when compared to other conventional schemes.***

***Keywords— fault tolerance, reliability, cloud computing, Decision making, Fine grained check pointing.***

## I. INTRODUCTION

Cloud computing is becoming a challenging task as its demand upsurges over worldwide. Also it grabs more attention due to its usage in variety of application areas. Still some of the people hesitant for using cloud in ther eal time applications Dhingra and Gupta (2016). Currently, several researchers are functioning under the cloud to contribute the power of cloud and its related remunerationstowards real time applications. In cloud computing, virtualization is the crucial model and it has quite a lot of challenges such like task scheduling, quality of service,efficient resource utilization and dynamic resource management. Supporting real time system on the cloud infrastructure is a significant part in today's world because of numerous usage of real time systems everywhere. The variety of applications are commencing fromsmall cell phones tobigger industrial controlsand as well as from mini pacemaker to superiornuclear plants. In that more number of system are safety critical systems that must be reliable. Generally, real time system bring up the information processing system that should react in a finite period for the outwardlyproduced input stimuli. The logical outcome is not only a greater concern for correctness but also its delivered time, Stankovic(1988). The acknowledgement failure is equally asworse as incorrect response, Kim (2002).There are two key features under this system that are separated byother general-purpose systems.

The fault tolerance and timeliness are the two features, Tsai, Shao, Sun and Elston(2010). The timeliness is meant for detecting the time limit of each real time task execution while fault tolerance meant for pinpointing operating condition even in the fault presence , Coenen and Hooman (1990).

The error occurrence ratio may be greaterviacloud infrastructurein real time applications since the nodes are distant from the transceiver. As many real time systems are safety critical systems, fault tolerance level is highly required, Reis,. Chang, Vachharajani,. Rangan and August (2005).The proper working of safety critical system is essential in order to evade failures causing a financialloss and also casualties, Kim (1984).Fault tolerance is referred as methodology either permitting the system design to carry out the task even though one of its portions drops futile or defining the system capacity to react quickly towards unexpected equipment or programming interruption.Instead of shutting down the system fully, perhaps fault tolerance solutionslet a system to work with lesser capacity.

This study was undertaken to handle the fault tolerance intricacy which is the major challenge overcloud computing infrastructure. So an adaptive fault reduction scheme is introduced. In preemptive cases, the scheme is influenced by user's requirements for the request andthe existing information of VMswhile scheduling. Also, the reliability outline of the node is increased employing adaptive fault tolerance techniques. The reliability of the node could be increased with the replication techniques that reduces the existence of failure nodes. Moreover, availability and the response time is considered for enhancing the performance of the cloud nodes. In this system, a fine grained check pointing and replication methodsare employed and also the finest suitable methods are chosen based on the present condition of cloud setup. Some challenges in cloud computing is shown below

- The replicas count is not either fixed or static which causes influence on the cloud. It happened due to other virtual machine that will also going to perform similar services. These VM are utilized for handling the request coming from other user services until now. Thus the the cloud will mislay lucrative charges.
- The implementation of replication technique on every single server was not cost-effective. The replication is required for the services that are applied to top most valuable virtual machine having an extreme influence on it. But the establishment of ultimate virtual machine was a bigger challenge.

- Finding interval length in check pointing is a bigger challenge under this scheme. The check points are terminated when there is a fixed interval range taking cloud resources thereby increasing.

**The main contribution of this work is given as**

a) To analyze and review various fault tolerance models in cloud environment.
b) To develop reliability framework of cloud nodes using adaptive fault tolerance techniques.
c) To implement backward recovery techniques using fine –grained checkpointing.
d) To minimize the probability of failure nodes using replication technique.
e) To enhance the performance of cloud nodes in terms of availability and response time.
f) To validate proposed framework with existing fault tolerance models.

The section 1 deliberates the introduction and the background of fault tolerance mechanism in cloud computing. Section 2 analyzed several traditional fault tolerance methods. The proposed methodology is deliberated in the section 3. Performance measure/ results is given in section 4.

## II. Related Work

### A. Existing Algorithms for fault tolerance mechanisms

Mesbahi, Rahmani and M. Hosseinzadeh(2018), deliberated the Reference Roadmap and its high availability and reliability in cloud computing infrastructure. The specific concern was given to every single step of reference road map. The twosignificant research gap was found in this reference roadmap. The load that has to be balanced in the distributed systems such like cloud computing which is the central concern of this work due to lack in traditional architectural design. As load balancing shows significant role in cloud environment, it is essential to consider it appropriate designing part which is studied in this paper, Mesbahi and Rahmani (2016). Additionally, the present load balancing solutions of cloud was characterized and it wascategorized as Architectural-based, Artificial Intelligence-based load balancing mechanisms and General Algorithm-based. After categorizing, these solutions were evaluated with suitable metrics and their benefits and drawbacks was also deliberated. The performance of the system was enhanced in the cloud computing systems through load balancing including energy consumption and stability as well. Kumari and P. Kaur(2018), presented a fault tolerance-related issues occurred in cloud computing. In this work, a significant impressions and architectural facts were emphasized. This work reviewed an existing fault tolerance techniques in cloud system that tolerates the central problems. The main intention of this work was to analyze the traditional fault tolerance approaches along with challenges required so far to be overcome. Moreover effective solutions from the promising techniques were found which recognizes research direction. Due to the demand in fault free cloud computing systems, this model concentrated on the set of proactive fault management techniques. In this case, numerous testing could be undergone on the cloud computing system to analyze the node which gets failed at every consecutive testing cycle. The fault in node was detected by applying Prediction algorithms or other specific classification algorithms that identifies the factors such as load and infrastructure affecting the node characteristics. From that collected information, the failure nodes were likely predicted. Thereafter, the faulty node was either eliminated or replaced with backup node which makes the cloudsystem to work efficiently. Thus the reliability of cloud computing system was improved and it also fairly performed the fault management job without any flaws, Gupta (2018). Though there are several faults in the cloud computing system, Byzantine fault detection is one of the significant challenge towards fault tolerance mechanism. The initial stage of this byzantine fault was difficult to detect and it was even propagated to nearby VMs before finding it. Hence most of the mission oriented applications were not operated under cloud due to its fault complexity. The applications like big data analytics that completely goes incorrect if the byzantine faults was not detected earlier. This is the reason for suggesting a previous work named as fool-proof Byzantine fault detection method which further enhanced in our work to elevate the design of fault detection system using scheduling algorithm (WSSS) and checkpoint optimization algorithm (TCC). These algorithm was efficient in terms of eliminating the Byzantine fault. From the data analysis it was proved that the WSSS and TCC algorithm was efficient by way of reducing the overhead and allocating efficient virtual machine, Chinnathambi, Santhanam, Rajarathinam and Senthilkuma (2018). The author examined the cloud virtualized system architecture i.e. HA Proxy in this paper, Roveicy and Bidgoli(2018), The two methods such as HA proxy and Amazon ELB was compared and studied to analyze each method which enhances how muchreliability and availability. The Amazon ELB was better than HA proxywith its effective server allocation strategy that reduces the latency. The relevant metrics such like amount of Request Count and average latency was taken into account to demonstrate high reliability and availability. The result revealed that the performance of Amazon ELB was superior than HA proxy concerning the factors such as processing time, connecting time, waiting time and fail requests. Several existing researches utilized static load balancing or response time of server for evaluating the capacity of load balancing. Perhaps there is a lack that causes uneven server load. The dynamic annexed balance method was suggested to resolve those issues. The server loading and the processing power was considered in cloud load balancing (CLB), therefore creating the cloud server system to meet all the computational requirements. The experiments are addressed with two CLB to prove the innovative proposed approach, Chen, Chen and S.- Kuo (2017).

### B. Approaches beneath Deadlock Detection

This paper Sood(2018), deliberates the SNA techniques, detecting the deadlock for fog layer resources of mobile device. The novel scheme named as free space fog has been proposed to collect the available free resource for allotted jobs to eliminate the deadlock. In the event deadlock was detected, some protocol was given to deadlock manager to increase the resource usage and decrease the response time in fog layer. The deadlock was also accomplished effectively with two distinct clouds i.e. private and public cloudsaside from fog layer and free space fog. The resources were provided through assigning the priorities of request from the cloud users. Thus this proposed framework was utilized to provide the QoS and also reliability to the users. As deadlock detection is the major impact of cloud computing system, it has to be detected in earlier. An unstructured deadlock could be detected with the use of gossip protocol over the cloud infrastructure. The special features of gossip protocol utilized in this scheme will make this deadlock detection algorithm as scalable, efficient, fault-tolerant, retaining safety and liveness properties. The proof of algorithm was also exposed. This proposed algorithm is having message complexity as$O(n)$, in which n represents node count. From the performance evaluation it was proved that this deadlock scheme was much efficient than the traditional deadlock scheme, Lim, Suhand Yu (2014). Nguyen, Dang, Pham and Nguyen (2015), studied an allocation of resources on infrastructure level rather studying on mapping of physical resources towards virtual resources over cloud computing environment. The novel algorithm was implemented in allocation of resources towards the infrastructure allocating the virtual machine dynamically amongst the cloud computing applications related to Deadlock detection and threshold method. Itcould be utilized for optimizing the decision of resource reallocation.The cloudsim simulator was used to check the performance of algorithm and from the experimental results it was verified that our approach detect the deadlock quickly and it also resolves it

Zhang, Liang, and Ahn,(2019) The issues occurring in the adaptive event triggered dynamic output feedback fuzzy control performing nonlinear networked control systems are explained briefly in the work. Actuator failure and packet dropouts are the two key factors which are considered concurrently. The systems considered are described with the help of the Takagi-Sugeno (TS) fuzzy model. The missing data are illustrated by employing the Bernoulli random distribution process. Then the malfunction occurring in the actuator is illustrated by adapting the actuator failure model. The computational resources are protected with the help of a novel adaptive event-triggered strategy. After that a fuzzy dynamic controller was constructed based on the Lyapunov stability theory, mainly for providing an assurance for stochastic stability and then H∞ performance are considered. Then at last, for exhibiting the effectiveness of the proposed control strategy the simulation results are delivered.

Ganesh, Sandhya and Shankar (2014),introduced a Takagi-Sugeno (T-S) fuzzy model which mainly concerns packet dropouts and actuator failure.In order to portray the fact of data missing, the Bernoulli random distribution methodwas employed. The failure of actuator was depicted through adopting stochastic function accompanyingMarkovian variables. Safeguarding computational resources is the significantchallenge that accomplished with innovative adaptive event-triggered mechanism was introduced. Under the outline of Lyapunov stability theory, the stochastic was guaranteed designing a fuzzy dynamic output feedback controller. Also H∞performance for considered schemes was assured. At last, the simulation results shows the proposed control strategy demonstrating its utility.

This section reviewed the pros and cons of the fault tolerance mechanisms with distinct approaches. The efficient fault tolerance mechanism was discerned in this section and it further extends to enhance the efficiency of the system with higher reliability and availability of cloud nodes.

## III. PROPOSED WORK

The proposed work and the work flow mechanisms are deliberated in this section. The novel work proposed in this cloud framework has many advantages which leads the system to work efficiently. Moreover, distinct parameters that are affecting the virtual machines was evaluated. In order to enhance the efficiency of proposed framework, the research gaps of cloud infrastructure are intensely observed in the literature section. The reliability architecture assessment of cloud virtual nodes was described in the flow diagram.

Initially, an input of virtual nodes are taken from the input buffer and the failure of the virtual machines was evaluated based on the following parameters:

- Result generation
- Turnaround time evaluation
- Reliability parameter verification.

In this proposed methodology, the core modules were reliability assessment algorithm and decision mechanism algorithmwhich enhances the reliability of virtual machines after every computation cycle.

This proposed scheme utilizing n Virtual Machines wherein each machine acquired its input from input buffer. All virtual machines concurrently acquires input from it. Every single machine takes the input, executes the application and produces result. Then the results are directed towards the Acceptance Computation module. If the acquired results are correct, then its computational time is compared with threshold time. If the computational time exceeds threshold time, the node becomes failure and its reliability is not tested. When it's computational time declinesor equal to the threshold time, this time VMs reliability has been checked.

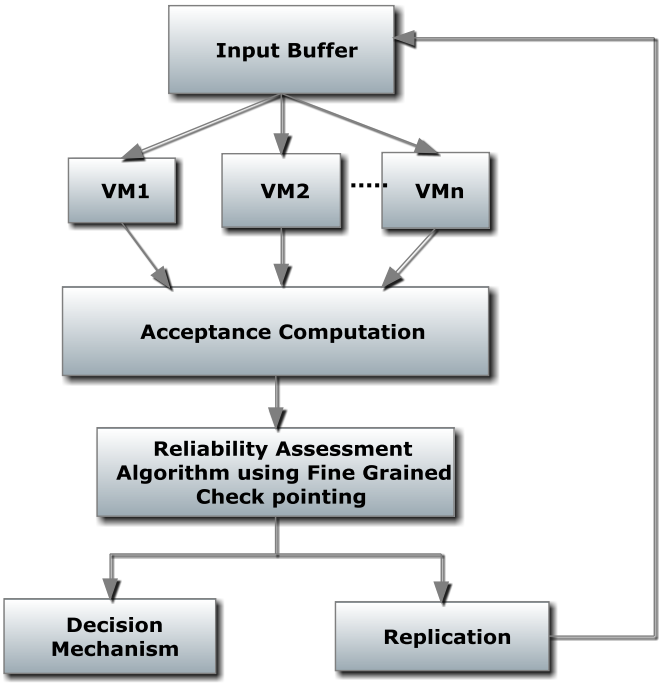


**Figure. 1 Reliability Assessment Architecture Dhingra and Gupta(2019)**

The virtual machine reliabilityis adaptive as it deviates for each computing cycle. The reliability of every single machine is always perfect at the beginning stage. At the time of managing the processing node, its reliability increases to acquire correct result within less time. In case of failure of making a correct processing node, its reliability increases. In such a case a Reliability Assessment Algorithm (RAA) is utilized for measuring its minimum and maximum reliability level. The RAA halts the furtherworking of node when there is a drop in the processing node below its minimum reliability. At that time thefine grained check pointing is used for recovering that node. However the replication technique will be used if the node reaches below the minimum reliability level more than three times. Afterwards, the replication technique is used which means the faulty node will be exchanged by the new node, this will be done by backward recovery. On the other hand, if the reliability level of the node gets satisfied reaching above the minimum reliability it will directed towards thedecision mechanism module that pick out the final output for a computing cycle. Thus the selected output node is having the highest reliability among the other competing nodes that produced the results within shorter time.If two highest reliability node, the smaller IP address prevailing in node is selected as output.System reliability level have to be considered at minimum level to allow the result. The best reliability is compared with system reliability level via DM and it has to be either greater than or equal to system reliability level. In case of not achieving the SRL with best reliability node, the failure signal is raised for the computation cycle or else it gets permitted.

### A. Reliability assessment algorithm

The minimum reliability level of the node is computed with this reliability assessment to find and replace the failure nodes. Also the fine-grained check pointing is implemented when the nodes get failed. If the node failure continues, replication method will be implemented for replacing the node using backward recovery method. Along with the reliability assessment algorithm, a decision mechanism module is also operated to evaluate the node reliability

**Table: 1 Reliability time assessment**

| VMID | Totaltime | Reliability Time | Result |
|---|---|---|---|
| 0 | 852 | 0.99734 | Pass |
| 1 | 114 | 1.994 | Pass |
| 2 | 366 | 2.992 | Pass |
| 3 | 954 | 3.9895 | Pass |
| 4 | 776 | 4.9869 | Pass |
| 5 | 407 | 5.984 | Pass |
| 6 | 747 | 6.981 | Pass |
| 7 | 562 | 7.979 | Fail |
| 8 | 134 | 8.976 | Fail |
| 9 | 137 | 9.9739 | Fail |

Virtual machine 0 to 9 is given in table. The total time and reliability time is checked for each virtual machine to identify whether the machine performing its task with expected reliability time. In the above table it is clearly shown that the 7, 8 and 9 failed to perform the task when its reliability time increases.

Algorithm 1: **Reliability Assessment Algorithm**
Input : *Firstly*reliability ← IR=1, n :=1 *from configuration* RF,

Step1: MR ← max Reliability
Step2: Mir← min Reliability
Step3: Ns ← node status
Step4: If Ns = pass then
Step5: IR= IR+(IR * RF)
Step6: If n > 1 then
Step7: n=n-1
Step8: End if
Step9: Else if Ns = Fail then
Step10: IR =IR –(IR*RF*n)
Step11: n=n+1
Step12: End else if
Step13: If IR >= MR then
Step14: IR = MR
Step15: End if
Step16: C ← count=1;
Step17: While c<=4
Step18: C+1
Step19: If IR < Mir then
Step20: Ns = dead
Step 21: call_proc: remove_this_node
Step 22: call_proc: add_new_node
End if
End while

Algorithm 1 depicts the reliability assessment algorithm in which the minimum and maximum reliability of nodes are identified for replacing the failure nodes. The node status is checked for reliability analysis to fix the specific nodes for execution eliminating the failure nodes. Step 3 shows node status, if the node status is passed it proceeds the process of IR= IR+(IR * RF).

### B. Decision mechanism algorithm

Among all those competitive node, output of the node is likely chosen with the maximum reliability under this decision mechanism. The several assumed parameters in this model is Acceptance Computation, Maximum Reliability Level, Minimum Reliability Level, Reliability factor and SRL (System Reliability Level).

**Algorithm 2: Decision Mechanism Algorithm**

**Input:**
Step 1: IR←*Initially* reliability=1, n =1
Step 2: RA ← Node Reliability
Step 3: BR ←best Reliability
Step 4: FR ← find reliability
Step 5: S ← Status
**Step 6: Input** *from configuration* SRL
Step 7: BR = FR of node with highest reliability
Step 8: If BR >= SRL
Step 9: S = Success
Step 10: End if
Step 11: Else if BR <= SRL
Step 12: S= Fail
Step 13: End if
Step 14: Else
Step 12: Perform_backword_Replication
Step 13**:** End

Algorithm 2 depicts decision mechanism algorithm. Step 6 in the algorithm 2 shows that eth input is taken from the configuration SRL. It checks forBR = FR of node that has highest reliability. If BR >= SRL, then status S gets succeeded
If BR <= SRL, its status gets failed. Or elsethe backward replication is performed back.

### C. Fine grained check point algorithm

Most of the grid computing systems uses check pointing as sensitive fault tolerance scheme to lessen the failure impacts. Moreover, the replication techniques has been implemented in massive number of cloud computing systems. From the server's perspective, the additional allocation of components is the major influence of this replication scheme in terms of cost handling for request replicas. But it is mostly useful for the request coming from the other users. On the other hand, user's perspective determines that this replication reduces waiting time of the componentswhich executes the replicas of other user request. The major benefit of this check pointing scheme over replication is protecting the cloud computing resources for other user's request and decreasing the profit losses with replication methods.

The two key factors such as check pointinglatency and interval is havinglarger impact over the check pointing algorithm whereas the check pointing interval represents the time between the checkpoint and consequent checkpoint and check pointing latency signifies the time taken for saving the checkpoint. There will be greater amount of checkpoints when there is a very small checkpoint interval. So the cloud resources are highly utilized whilst saving the checkpoints and therefore the check point latency is largely produced. Additionally, the request has to be recomputed if there is any failure occurs. Here the cloud resources are weakly utilized with the fewer amount of checkpoints while saving it, thus the result of check pointing latency is lower. Consequently, Checkpoint interval length is a vital challenge in check pointing mechanism.

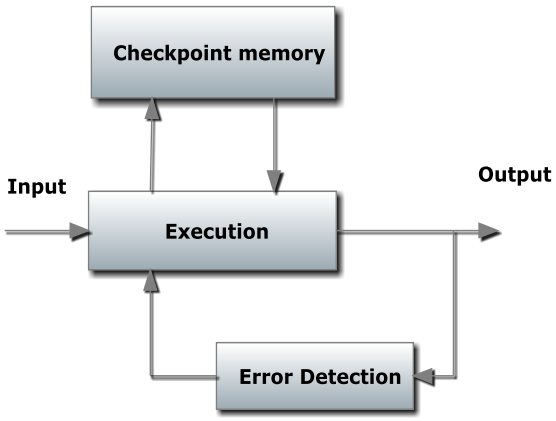


**Figure. 2 Logical Outcome of checkpoint memory**

The checkpoints are terminated at fixed interval which makes the cloud resources to beutilized that subsequently risesthe checkpoint latency. Thus the objective of this paper is to enhance the algorithm with the capability of determining the check pointing interval length adaptively premised on the VM's failure history. The checkpoint interval length assumed by our fine grained checkpoint algorithm is not fixed in the course of executingusers requested job.The subsequent check pointing interval is only allowed to be calculatedwhen executing the present checkpoint.This calculation is made concerning the failure history of virtual machine. The algorithm will minimise the checkpoint intervals once there is a poor failure history and lengthens it if there is a good checkpoint intervals.

**Algorithm 3: Fine –grained check point Algorithm**

Input: R← Reliability time
VM ← Virtual machine
T ← thresholds time
NC ← number of count=4
N ← Number count =1
$B_{TF}$← False
$B_{Ts}$← True
Step 1: While (!Task execution completed) do
Step 2: If (R>=T)then
Step 3: Estimation();
Step 4: END IF
Step 5: if(R <= T)
Step 6: R++;
Step 6: N++;
Step 7: If (N<=NC)
Step 8: then $B_{TF}$ = true
Step 9: update the VM failure information.
Step 10: Create New VM

Step 10: end if
Step 11: end while

Replication Algorithm
VM ← Virtual Machine
R ← Reliability time
MR ← Minimum Virtual Machine creation time
C ← Virtual Machine creation time
While (VM) do
If(R< MR && C<MC)
Find Best VM
End if
Else
Return Virtual machine creation page
End
End while

Algorithm 3 portrays fine grained check point algorithm. This algorithm is implemented for checking and the enhancing a little bit of reliability time of VM for completing the particular task efficiently. If the execution of the task halts suddenly, here reliability is checked on relating it with threshold values. Just In case of greater reliability than threshold time or equal, then there will be no problem in finishing the task at scheduled time without interruption. IfR<= T, the reliability and node count 4 is increased. If the N<=NC thus far, then BTF =true.
Then the VM failure information is updated.

## IV. Performance Analysis

In this section, the performance measure was evaluated to prove the efficiency of proposed adaptive fault tolerance mechanism. The performance of the framework was evaluated in terms of execution time. The reliability assessment, decision making and the Fine grained check point algorithm efficiency was provedwith the outcome of these measures. To monitor the correctness of results at initial level, App Dynamics is utilized i.e. at the Acceptance Computational Module. The timing of result produced by each virtual machine will be monitored with Watch Dog Timer Software. Amazon EC2 and Microsoft Azure are the rental based infrastructures to meet the replication technique. ProActive grid interface is also available at Amazon EC2 cloud to run the real time applications.

Amazon Web Services provides a platform that is ideally suited for building fault-tolerant software systems. Among the several accessible cloud simulator domains, Cloudsim is performing well. The classes of the package permit the development of algorithms based on fault tolerance which in turn can supervise virtual nodes towards the identification of failures and then later resolves them. This provision can deploy both fine grained check pointing and replication mechanisms. This simulator offers the capability to calculate availability, throughput, time overhead and monetary cost overhead.

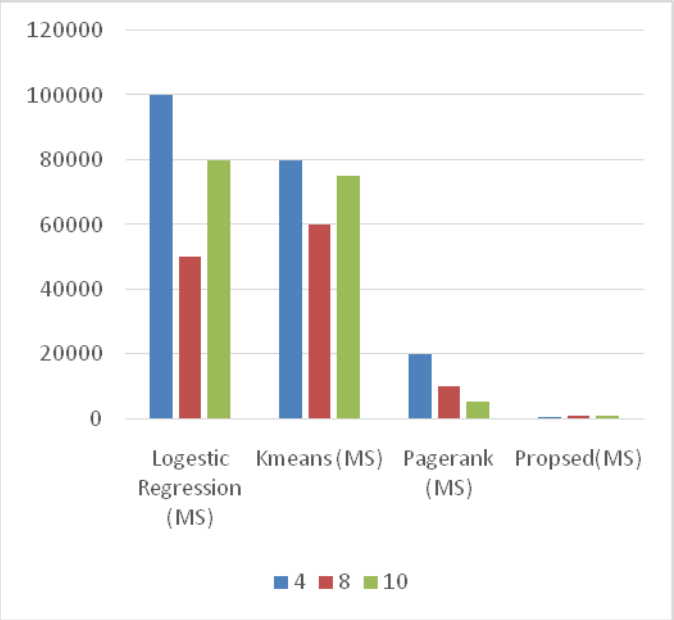


**Figure. 3: Execution time of existing and proposed method**

Figure 3 depicts the execution time of existing and proposed framework. The proposed framework accomplished better lesser execution while comparing with other methods. From the graph it was clearly understood that the proposed system takes only lesser execution time which makes the system more efficient. Here 4, 8 and 10 are the number of virtual machines taken into consideration for comparison. An existing logistic regression, k means and pagerank are compared with proposed scheme to show the superiority of proposed method.

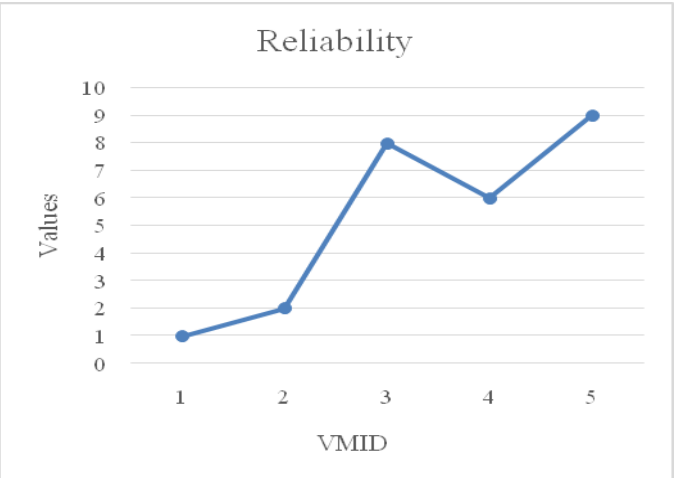


**Figure. 4: Reliability of virtual machine**

Figure 4 depicts the reliability of virtual machine 1,2,3,4 and 5. The VMID is given in x axis and reliability values is given in y axis. The reliability values of virtual machine is the significant part in fault tolerance techniques.

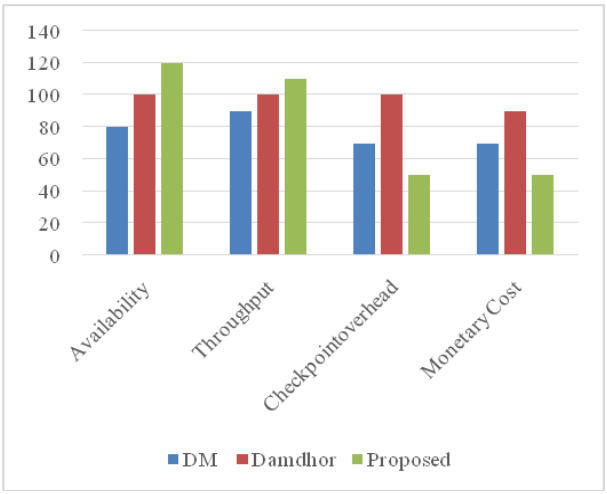


**Figure: 5 Comparison of existing and proposed methodology**

Figure 5 portrays the comparison of existing and proposed methodology. The availability and the throughput of the proposed method is accomplished better than the other existing approaches such as DM, Damdhor. The checkpoint overhead and monetary cost is lesser than the proposed methodology which enhances the efficiency of tolerating faults.

Simulation output:

The simulation screenshots of this fault tolerance mechanism under reliability frameworkis given below to prove the efficiency of this reliability framework, virtual machine creation, and Acceptance computation, Reliability Framework using fault tolerance technique and Reliability Framework for cloud virtual nodes. Figure 7 shows the random creation of virtual machine. An acceptance computation of reliability is shown in figure 8. Fig 9 and 10 shows the Reliability Framework for cloud virtual nodes.Initially the file is chosen and VM is created is based on that file. The optimized VM is computed based on the time constraints under the acceptance computation. Each VM has its own reliability time and its time will be enhanced using fine grained check point algorithm. The VM creation time, reliability framework are the parameters concerned for the decision making of the best VM. If the best VM is not selected for three times then it goes back to the initial mechanism

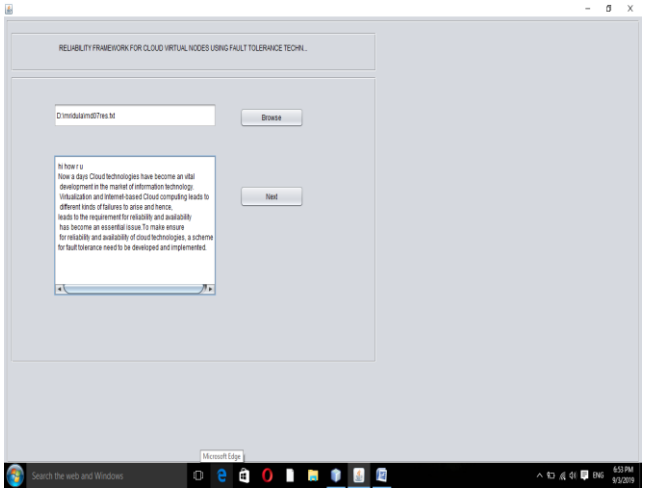

**Figure 6: Reliability framework**

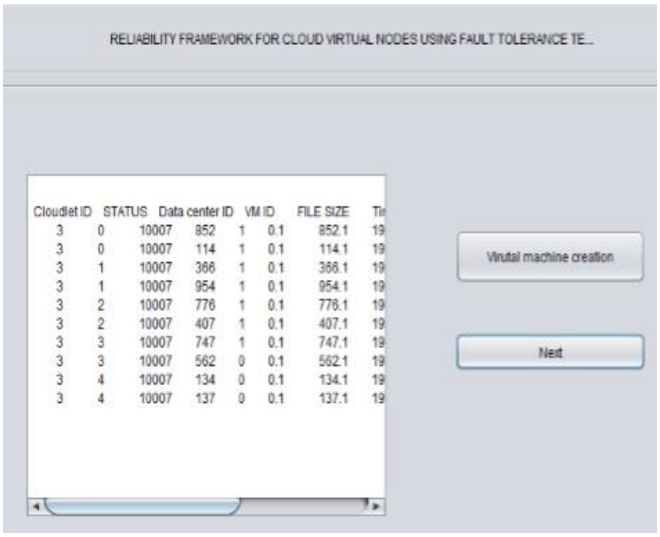


**Figure 7: Virtual machine Creation**

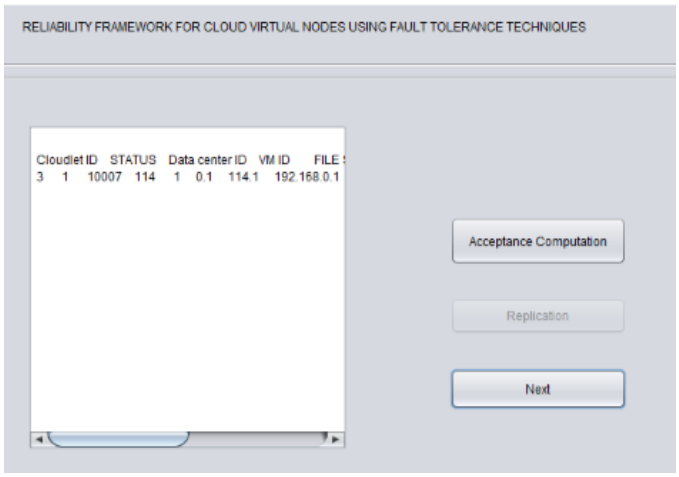


**Figure 8: Acceptance computation**

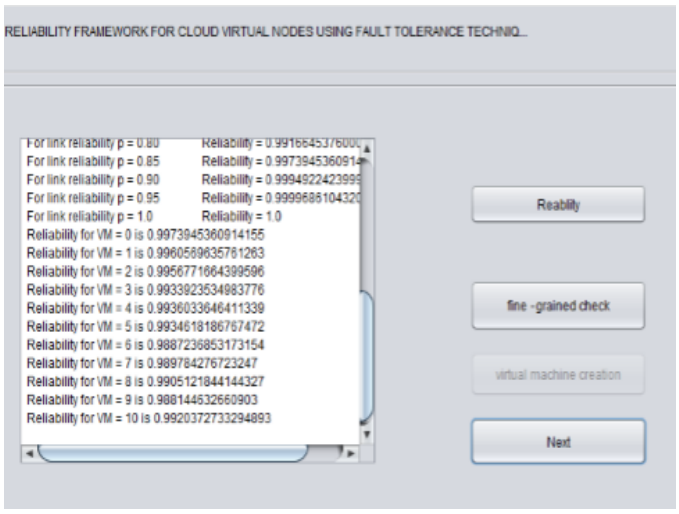


**Figure 9: Reliability Framework using fault tolerance technique**

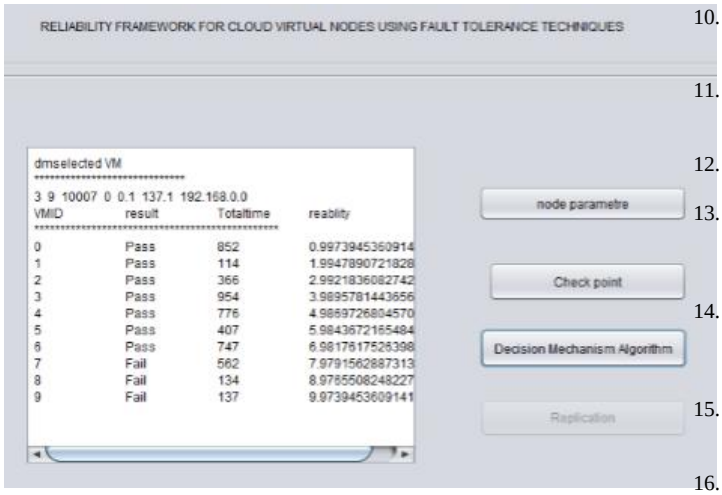


**Figure 10: Reliability Framework for cloud virtual nodes**

## V. CONCLUSION

To handle the major challenge of tolerating fault related complexity, a study was done over cloud computing and introduced fault reduction scheme. In pre-emptive cases, the requests from users were influenced by this scheme for scheduling VMs. Also increment in using adaptive fault tolerance techniques provides reliability to the node outline. The reliability can also be increased by using replication methods which lowers node failures. The response time and availability is taken into account for performance enhancement. In this system, replication and check pointing methods are employed based on present situation of cloud infrastructure. The execution time of the proposed method was superior to existing method with excellent outcome difference.

## REFERENCES


1. M. Dhingra and N. Gupta, “Comparative analysis of fault tolerance models and their challenges in cloud computing”, in International Journal of Engineering and Technology , vol 6(2), 2017, pp.36–40.
2. M. Dhingra and N. Gupta, “Architectural Framework for Cloud Reliability Model using Fault Tolerance Techniques”, in International Journal of Innovative Technology and Exploring Engineering,vol 8(11),2019,pp.515-519.
3. J. A. Stankovic, "Misconceptions about real-time computing: A serious problem for next-generation systems", Computer, vol. 21,1988, pp. 10-19.
4. K. Kim, "Toward Integration of Major Design Techniques for Real-Time Fault-Tolerant Computer Systems", Journal of Integrated Design and Process Science,2002 vol. 6, pp. 83-101.
5. W.-T. Tsai, Q. Shao, X. Sun, and J. Elston, "Real-time service-oriented cloud computing", in 2010 6th World Congress on Services, 2010, pp. 473-478.
6. J. Coenen and J. Hooman, "A formal approach to fault-tolerance in distributed real-time systems", in Proceedings of the 4th workshop on ACM SIGOPS European workshop, 1990, pp. 1-4.
7. G. A. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. I. August, "SWIFT: Software implemented fault tolerance," in Proceedings of the international symposium on Code generation and optimization, 2005, pp. 243-254.
8. K. Kim, "Distributed Execution of Recovery Blocks: An Approach to Uniform Treatment of Hardware and Software Faults," in ICDCS, 1984, pp. 526-532.
9. M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Reliability and high availability in cloud computing environments: a reference roadmap," Human-centric Computing and Information Sciences, 2018, vol. 8, p. 20.
10. M. Mesbahi and A. M. Rahmani, "Load balancing in cloud computing: a state of the art survey," Int. J. Mod. Educ. Comput. Sci, 2016, vol. 8, p. 64.
11. P. Kumari and P. Kaur, "A survey of fault tolerance in cloud computing," Journal of King Saud University-Computer and Information Sciences, 2018.
12. A. Gupta, "Machine learning based approach for fault tolerance in cloud computing," 2018.
13. S. Chinnathambi, A. Santhanam, J. Rajarathinam, and M. Senthilkumar, "Scheduling and checkpointing optimization algorithm for Byzantine fault tolerance in cloud clusters," Cluster Computing, 2018, pp. 1-14.
14. M. R. Roveicy and A. M. Bidgoli, "Comparative Analysis of Fault Tolerance Techniques in Cloud Computing: A Case of Armangerayan Co," International Journal of Engineering & Technology, 2018, vol. 7, pp. 432-436.
15. S.-L. Chen, Y.-Y. Chen, and S.-H. Kuo, "CLB: A novel load balancing architecture and algorithm for cloud services," Computers & Electrical Engineering, 2017, vol. 58, pp. 154-160.
16. S. K. Sood, "SNA based QoS and reliability in fog and cloud framework," *World Wide Web*, vol. 21, 2018, pp. 1601-1616.
17. J. Lim, T. Suh, and H. Yu, "Unstructured deadlock detection technique with scalability and complexity-efficiency in clouds," International Journal of Communication Systems, 2014, vol. 27, pp. 852-870.
18. H. H. C. Nguyen, H. V. Dang, N. M. N. Pham, and T. T. Nguyen, "Deadlock detection for resource allocation in heterogeneous distributed platforms," in Recent Advances in Information and Communication Technology 2015, ed: Springer, 2015, pp. 285-295.
19. Z. Zhang, H. Liang, C. Wu, and C. K. Ahn, "Adaptive event-triggered output feedback fuzzy control for nonlinear networked systems with packet dropouts and actuator failure," IEEE Transactions on Fuzzy Systems, 2019.
20. A. Ganesh, M. Sandhya, and S. Shankar, "A study on fault tolerance methods in cloud computing," in 2014 IEEE International Advance Computing Conference (IACC), 2014, pp. 844-849.


## AUTHORS PROFILE

**Mridula Dhingra** is a Research Scholar at Faculty of Computer Applications at Manav Rachna International Institute of Research and Studies, Faridabad campus. She has completed her Masters in Computer Applications from Maharishi Dayanand University, Rohtak. She has total of 12+ year of experience in teaching and research. She has authored and coauthored 4 research papers in various National and International Journals.

**Dr. Neha Gupta** has completed her PhD from Manav Rachna International University and has total of 14+ year of experience in teaching and research. She is a Life Member of ACM CSTA, Tech Republic and Professional Member of IEEE. She has authored and coauthored 34 research papers Journals (Scopus indexed) and IEEE/IET Conference proceedings in areas of Web Content Mining, Mobile Computing, and Cloud Computing. She has published books with publishers like IGI Global & Pacific Book International and has also authored book chapters with Elsevier, CRC Press and IGI global USA. Her research interests include ICT in Rural Development, Web Content Mining, Cloud Computing, Data Mining and NoSQL Databases. She is a technical programme committee (TPC) member in various conferences across globe. She is an active reviewer for International Journal of Computer and Information Technology and in various IEEE Conferences around the world.