INTEGRATED TRENATE FAULT TOLERANCE OPTIMIZATION FOR RESILIENCE IN CLOUD COMPUTING ENVIRONMENTS

Anwar Ahamed Shaikh[1] Dr Shish Ahmed[2]

[1]Department Of Computer Science and engineering,

Integral University Kursi Road, Lucknow, Uttar Pradesh 226026

[2]Department Of Computer Science and engineering,

Integral University Kursi Road, Lucknow, Uttar Pradesh 226026

Abstract: Cloud computing is one of the most trending model which is part of the modern IT industries either it is large scale or smaller unit in which services are provided over the global network that is internet. As we know cloud computing provide the resources are dynamic in nature, so the fault tolerance system is the demand of time that if one or more faults occurs in any components the supply of services must be continued and delivery of services must be sure . Fault tolerance is a robust mechanism in cloud computing which act smartly to overcome any failure occurred due to hardware or software failure and it assure the user to provide the availability of resources at satisfactory level. The expanding interest for adaptability of cloud computing over the globe autonomous way, and effectiveness in facilitating applications without the weight of establishment this is normal that various kind of failure occurs. While there are various approaches and technics do deal with fault but even though fail to deal the truncation of network route and overburden of Virtual machine that reduce the throughput. In this research paper by the use of a Ternate Fault Tolerance technics the resilience of cloud computing improved. It test the success of the active virtual node and maintain the network path by Escalate techniques, and assure the availability of virtual machine through the eternal alacrity in which the reserve memory used for technological use. Overall in this research paper we propose a solution where allocation of the task on the basis of success rate of virtual machine which improve the throughput and maintain the network path. Thus the Ternate fault tolerance improve the resilience of cloud computing environment with better throughput.

Keywords: Cloud computing, fault tolerance, reliability factor, virtualization, fault handler, check-pointing, updating virtual machine.

1. INTRODUCTION

As cloud computing is growing is such a way that it becoming the integral part of IT industries where network infrastructure, computing resources such as computer hardware, operating system, networks, storage, computing are available instantly on-demand as pay per use [2]. Either it is small scale or large IT industry Cloud computing become more popular which provide IT infrastructure and computing services to the end user such as storage, network, bandwidth and applications [1]. As cloud computing is getting more adoptable the news challenges arises.

The essential characteristics of cloud computing are

- One demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

With help of different service models and deployment models, all the on demand services provisioned to the clients by cloud computing providers. As we know that complex application in huge number run over internet through various nodes in the form of Virtual machines [8]. If there is one or more fault occurs during execution of application or updating, it coast more in the form of time and power and it will also a setback to the of cloud service provider's reputation.

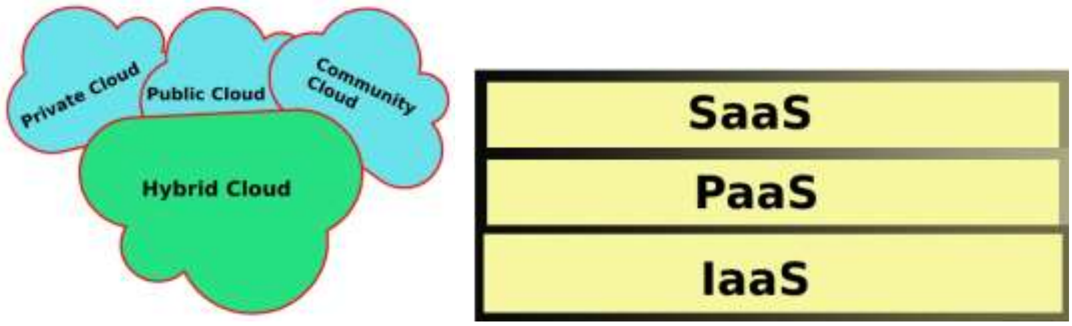

Image: 1 Type of Cloud and their services

Due to dynamic fluctuation of demand of resources by users so the management and monitoring of resource availability through virtual machines is necessary to supply the quality of services and assure the guaranteed and reliable delivery of resources as it is key part of the service level agreement, so the fault tolerant system is demand of time which manage the and fulfilled the services to the user if any fault occurs during the execution of application or computing [6].

Achieving high availability is not normal as it's known that in cloud computing environment there are hundreds to thousands virtual machines as well as physical servers participated in providing resources to clients over network. Due to the dynamic variation in the demand and upgradation in various components there are more chances of occurrence of fault in on or more components. To prevent these mishaps, some robust mechanism should be part of the system to handle the possible failures [32]. Recreating replica of the existing virtual machines will definitely make it sure that the supply and availability of resources as backup, but survey done by Computer failure data repository (CFDR) show that as the number of Virtual machine get increased the chances of failure also increase [9].

In Cloud computing three level of Service models [13] (a) SaaS (Software as a Service): all cloud service providers like Amazon web services (AWS), Salesforce, Microsoft Azure provide software applications to clients. (b) PaaS (Platform as a Service): for development of applications, testing of software, managing of application big cloud player provides cloud-based platforms. (c) IaaS (Infrastructure as a Service): It is a form of cloud computing that provides users computing, network, and storage resources over the internet, on a pay-as-you-go basis. It enables clients to manage the resources according to requirements in variable workloads environment [12]. Research concerns, like as security and fault tolerance [3] are fully managed in the cloud technology. Fault tolerance is one of the key issues in the cloud; it refers to all of the approaches that enable a cloud system to adopt the software faults that remain in after its developments [4]. In cloud computing system faults either handle by reactive fault tolerant approach or it handle by proactive fault tolerant approach, the major advantages of fault tolerance in cloud computing is to recovery from any failure, improvement in performance and cost reduction [11]. In 2011 cloud services provide by Microsoft face outage approximately 150 minutes, in early of 2014, Gmail was down for nearly 50 minutes and in year 2013 Facebook was also facing difficulties in uploading the photos and others key activities [7]. In the same year a key service known as google Docs by Google face difficulties due to some faults in memory during the some software updates. In ending of 2012 Amazon web services was down for more than 6 hours and the same year one of the dominant domain service providing organization Go Daddy was down due to some failure and due to this millions of website were facing issues [1]. Researcher proposed many fault tolerance techniques to handle the effect of fault but in this paper we proposed a novel technique that is Optimize ternate fault tolerance for resilience in cloud computing environments which will improve the resource availability and handle the faults in efficient way. This paper proposed a novel technique which analyze the success of the virtual nodes and those node failed to achieve the par level will remove from the list through which the chances of failure of virtual machine reduce and this way it will improve the throughput of system, by maintaining the network path through escalate lane technique which create a ruler node which maintain the network path when the destination node get failed, and by use of the reserve memory during the software upgradation the service would not stopped and overall performance will surely degrade but not going to shut. This paper has 5 sections, in section 2 the related work had discussed, in section 3 the propose model description detail is discussed, while in part 4 present the performance of the proposed model and its comparison and in last section the conclusion and the future work is discussed.

2. LITERATURE SURVEY

To handle fault in cloud computing environment number of authors proposed different approaches and analysis [10, 14, 15, 16, 17, 18, 19]. To provide the high availability and guaranteed resource in optimized Fault tolerance system a very few research have been done. With help of user based API and using Virtual machines to detect the fault for particular cloud model, many researcher addressed an optimize way to handle the Fault in cloud environment. Focusing on certain framework and delivery models, T Chana et al. [18] establish the implementation of Fault tolerance in Cloud by focusing on independent recovery. They show that when fault handle with collaborations especially by provider and end user who face the problem due to occurring of fault its returns better outcome. With help of a prototype implementation they perform the practical to know the outcomes of collaborative approach to handle the fault. Maloney et al. [21] after monitoring many FT approaches they proposed that rollback method to recover from failure in long running application will be best suited approach. To get better FT in cloud by using of rollback recovery model the need of proper management of system activity update required, where activity of failure node minutely observed and when required last successful point will be useful in recovery. By use of check-pointing and restoring techniques Kim et al. [22] proposed a model which show that the running node failure chances are much lower than the nodes which are idle or not. Addressing software glitches, Chen et al. [28] presented a software based fault handling mechanism, called SHelp, which have the very sharp and effectively programmed that recover from many types of software bugs available in the cloud environment. This model has a 'weighted' recovery point technique that survives software failures through bypassing the faulty path in a very effective manner. Qiang et al. [24] presents multi-level FT system for distributed application in cloud environment, in which such arrangement available to store all the activity or backup every running application states based on snapshot logic after some fix time periodically.

B. MOHAMMED ET AL. [32] also present a multi-component level recovery system which emphasize the generation of new virtual machines and rescheduling of process which enable multi component level recovery that provide the better and effective FT mechanism. Addressing high availability Jung et al. [25] proposed a broader solution of this problem of maintaining and ensuring the high availability and guaranteed of resources by reverse process of software to restore the previous successful process stage whenever any fault occurs. By proper component placement and load sharing by using of the information of the flow of running process. Agbaria and Friedman [26] proposed a study in which they suggested that instead of backing up all the state of running process only capturing the state at VM when restart required the saved state of virtual machine updated as the new copy of VM and process continue from the saved state. Another solution proposed by Shen et al. [27] in which provisioning of resource on demand mechanism will be use with API for the information to data centers where availability of computing resources reschedule on running mode according to the need of user also known as dynamic provisioning. This mechanism work at individual level service that's why it has control over the availability.

Zhou et al [30], proposed a checkpoint-based FT approach. An optimum checkpoint technique introduced in which server recovery based on the checkpoint image states. It has three level by which the certainty of resource of over the network, while again and again restart of failed service from the last successful checkpoint image state is time consuming process and also put an extra overhead on the CPU. Ray et al [23], proposed a proactive model based on understanding and coordination of various cloud players known as cloud federation, in which all cloud service providers share their unused resources such as VMs storage with others. In this a proactive fault tolerance model is acted based on the CPU heat which stop failure within the group. ILP and performance based algorithms are used to distribute the VM in federation proactive fault tolerance technique. Rogn et al [29], proposed a fault tolerance model based on replication. In this approach Virtual machines states running with various applications has to be captured on regular basis (check pointing) creating the replica of whole running application of the virtual machine ensure the high availability. According to the exhaustive survey an asymmetric virtual machine replication put burden due to skewed load balancing, and suffer from high check pointing involvement, delay in network, and put huge overhead on CPU resource. By considering all these kind of faults in virtual machines, networks and availability a novel framework should be developed so that it could implemented in cloud environment. The proposed methods will contribute to fault tolerance in cloud, with high throughput and increase the availability of virtual machines. The approaches that are already in use in fault tolerance are explained above. The next section explains the techniques and benefits in the proposed method.

3. PROPOSED ITFT MODEL

Fault handling and service dependability improve in Cloud computing systems by use of different approaches based on replication of virtual machines as well as checkpoint recovery system. Recreation of virtual machine to improve the efficiency, as it provide the backup in recovery during the failure of application run over virtual machine in the form of nodes. As we know that huge number of applications and services run over the numerous Virtual machines created from the physical servers on internet. If one or more faults reports and failure happen in running applications it will cost in the term of time money and the reputation of cloud service provider. Existing work like virtualized fault tolerance and adaptive fault tolerance techniques [34] handle the faults but still fail to maintain the network path and non-availability of virtual machine during system upgradation. To deal the issues in cloud environment an innovated robust framework should be developed in such a way which handle the fault and provide the guaranteed resource as discussed in SLA. Creating virtual machine for the backup surely improve the system but creating numerous VM also generate CPU overhead and probability of failure of VM will also increase.

The proposed model tolerates the faults on the basis of reliability factor (R_f) of the virtual machines. Only those computing node selected for assigning task whose reliability factor is high, and if the virtual machines not perform well on a real time applications the reliability factor of those machines reduce and it will remove from the list of active virtual machines.

The proposed model comprises of three key module known as Cloud Service Manager (CSM), Virtual machine fault handler (VFH) and Node Selection Mechanism (NSM) each module interconnected with each other and make it sure to provide a robust resilience mechanism to the cloud environments. The model name is as Integrated Ternate Fault Tolerance Optimization in Cloud Computing (ITFOC) fig.2.

3.1 Optimize Cloud Service Manager (OCSM) is integrated part of cloud architecture, and responsible to manage virtualization with help of hypervisor. Hypervisor is a low level program which create the virtual machines from the available physical server resources. Number of virtual machines can be created from a single physical server, during the creation of virtual machines all the records such as which virtual machine created from which server maintain by Hypervisor. A table know as Cloud Activity Record update the records like server id, VM id as well as the reliability factor (R_f) for identification of virtual machines. Activity table keep records which server's virtual machines assigned task many times and produce successful results. **R_f** is reliability factor of physical server machines places in data center whose resources are getting use for virtualization.

3.1.1 Load Balancer: On the basis of load balancing algorithms, load balancer assign the tasks or distribute the loads to those virtual nodes whose belonging physical server reliability factor is at satisfied level. Load balancer is integral part of OCSM which is connected with the others module and act according to the findings. A controller module grouped with FTDaemon active with each node, which act immediately when a node going to fail. It manage the migration of the job from faulty node to highest reliable node from the list of available active nodes [33].

3.2 Virtual Machine Fault Handler (VFH): When any virtual node look faulty due to some software initiated issue of running application or due to the fault in corresponding physical server the activity record table get updated. VFH vary its handling approach according to the type of fault, if the availability of resources delay due to some software update it use the reserve Cache recovery memory to migrate the process. If there is network path disturbance due to destination node failure the ruler node activated from the reserve cache where all the success checkpoint created Ruler node has all record of the destination nodes by use of FsImage techniques and replace the destination node until the backward recovery happened. VFH also use the check-pointing and restart mechanism and if fault handling done then the virtual machine of corresponding server will be available for further request.

3.3 Node Selection Mechanism: This module integrated with cloud computing architecture and has some key submodules such as **Status Flag (SF), Job Deadline Check (JDC), Cloud Activity Record Table (CAR)** and **Final Adjudication (FA)**. Status flag checks the status of the each virtual node and if it is ok then completion of assigned job deadline check by the JDC. If both the SF and JDC show ok then the reliability factor of the particular node increased and update in Cloud Activity Record, and then pass to final adjudication, if both of SF and JDC got failed then the node not forward to Final Adjudication rather it transfer to the virtual machine fault handler (VFH) which detect the fault and start recovery by past successful checkpoint. In a situation if algorithm run on the particular virtual machine produce proper result its SF is Ok but the produce result not done within time then the reliability factor of the node is decrease and updated in CAR and this particular VM records not move forward to final adjudication. Another situation where algorithm run in VM properly within the time but it may be possible that the threshold reliability factor still not met this time whole computing cycle failed and backward recovery start to handle the fault. The FA has only virtual machines that have successfully complete running applications within deadline time, nodes with highest reliability factor R_f chosen and successful checkpoint created [5] [31].

The working mechanism of this model is suppose we have generate n virtual machine from the different servers placed into data centers, with help of load balancer algorithms task assign to running virtual machines. Each VM running different real time application on them in such a way VM 1 run application X1, Vm2 run algorithm X2 and so on algorithm run on VM n is Xn, initially we assume all VMs that is n are running, if running applications or algorithms of one or more virtual machines successfully complete then the Status flag of all such VMs set or ok.

One key benefit of running different algorithm is system coincident failure chances are almost null, after set status flag Job deadline check (JDC) module check all of the complete job finishing time. On the basis of success SF and JDC, reliability factor (**R_f**) of each node calculated and reassign to them as well as will update these values to Cloud activity record, finally all these **R_f** passed to Final Adjudication (FA) which pick all those nodes having highest reliability factor and passed these selected VMs to service providers.

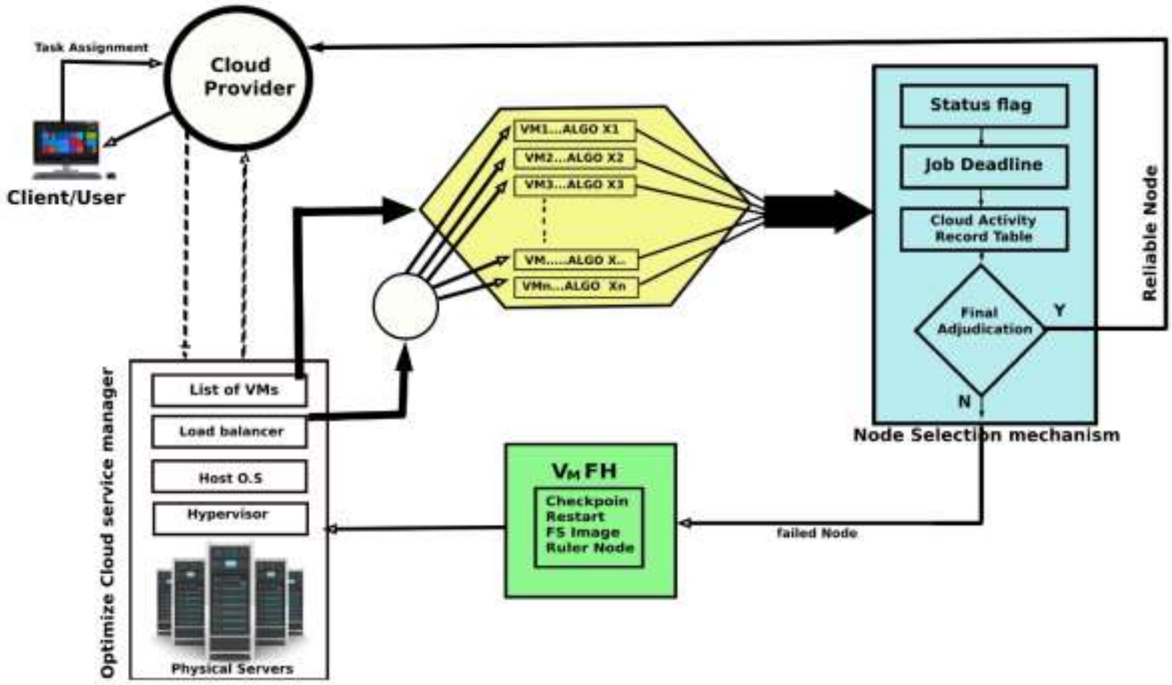

Fig. 2 Proposed Model (Integrated Ternate Fault Tolerance Optimization)

3.3.1 Reliability factor (R_f) Computation:

- n is number of successful task completion by a virtual machine
- m is number of task assign to a Virtual node
- maxR_f is ideal condition and minR_f is threshold reliability factor
- a virtual node cycle is successful if SF=1 and JDC=1

Algorithm

Algorithm

1. **if (SF=1 and JDC=1)**
2. **then n= n+1**
3. **m= m+1**
4. **Else**
5. **m= m+1**
6. **R_f = R_f + 1/mn**
7. **CAR updated**
8. **If(n=m)➔ (R_f =1)➔ maxR_f**
9. **If(R_f < minR_f)**
10. **Then model alarm the failure of node record updated in CART**

Adjudication Algorithm:

- **Final Adjudication has all the nodes whose SF=1 and JDC=1**
- **K is number of virtual machines whose SF=1 and JDC=1 where k<n for all nodes having R_f >= min R_f**

1. **Arrange all the R_f in non-increasing order**
2. **If(all R_f < min R_f)**
3. **Then All Node cycle failed and transfer to VFH where Load balancer informed not to assign any task to corresponding physical server. Recreation of node start**
4. **Else**
5. **For(i=1;i<=k;i++)**
6. **Virtual node forwarded to cloud provider and check point created in reserve recovery cache.**

4. RESULT AND DISCUSSION

In our proposed Integrated Ternate Fault Tolerance Optimization in Cloud Computing (ITFOC) use reliability factor computation for every virtual machine. Initially we assume **R_f =0.5.** Algorithm consider input as **minR_f** and input from Cloud activity Record Table (CART) {**R_f, n, m, VM_id, Ser_id }.** We discussed that the **R_f =1** is ideal condition, the **R_f** never become 0 because load balancer never assign task to the nodes having reliability factor zero.

4.1 Reliability Scenario:

Here we are comparing the different scenario of the reliability factors of our proposed model (ITFTO) with VFT [35], for 50 computing Cycles. Initially we assumed that the reliability factor is 0.5, figure 3 show in each cycle VM-1 is the continuous reliable machine, while figure 4 shows that VM-2 is continuously failing for every cycle, figure 5 show that the Vm-3 reliable for first 25 cycles and then continuously fail for remaining 25 cycles, figure 6 shows that the Vm-4 getting fail for first 25 cycles and the reliable continuously for the remaining 25 cycles. Vm-1 reliability factor increase after 50 cycles is 0.981 while reliability factor of vm-2 after 50 cycles goes to 0.048.While Vm-3 and Vm-4 reliability factors after 50 cycles are .501 and .520 respectively for the proposed model. We assumed that the initial reliability factor of all VMs are 0.5 and now if we assumed that the all VMs belong to different physical servers then it shows that the reliability factor increment is more than the decrements. After observation of fig 3, 4, 5, 6 it's evident that the proposed model perform better than the VFT model fault handling.

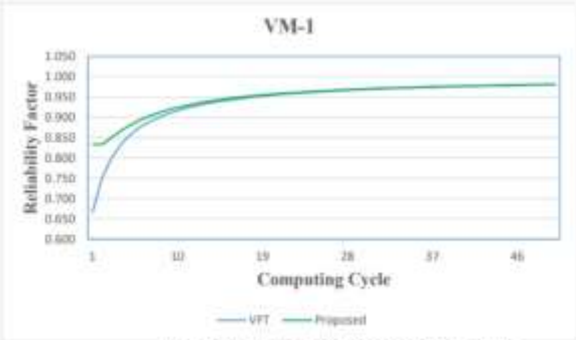

Fig 3 (a). Continuous Reliable

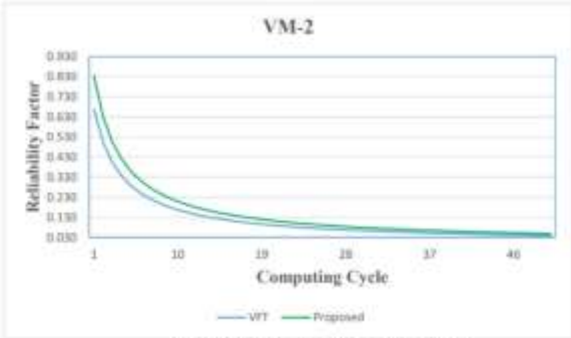

Fig 3 (b). Continuous fail

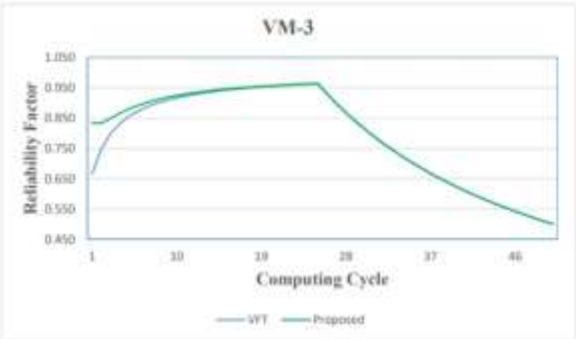

Fig. 3 (c) Reliable to failing

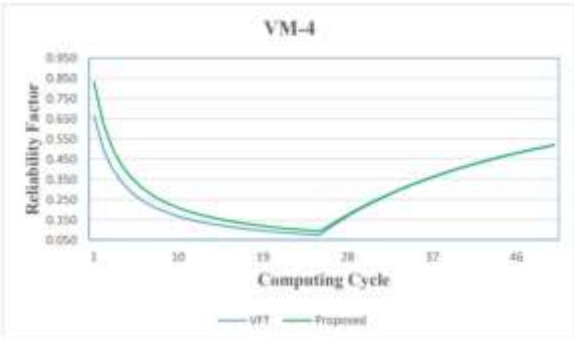

Fig. 3 (d) Failing to reliable

Fig 3. Simulation result of different scenario

4.2 Simulation Result: This work has been implemented through the Cloudsim 3.0 simulator and NetBeans IDE 8.1. Here we created 3 Virtual machine and assigned them different application.

Cycle	Job deadline	VM-1				VM-2				VM-3			
		SF	JDC	Reliability Factor	Finishing Time	SF	JDC	Reliability Factor	Finishing Time	SF	JDC	Reliability Factor	Finishing Time
Start	--	-	-	0.5	--	--	--	0.5	--	--	--	0.5	--
1	1700	1	1	0.833	1500	1	1	0.883	1501	1	1	0.833	1510
2	1550	1	1	0.833	1501	1	1	0.883	1550	1	0	0.625	1555
3	1601	1	1	0.85	1502	1	0	0.667	1603	1	0	0.5	1605
4	1900	1	1	0.867	1550	1	1	0.708	1600	1	0	0.417	1950
5	1700	1	0	0.743	1701	1	0	0.607	1701	1	0	0.357	1703
6	1800	1	1	0.771	1503	1	1	0.65	1502	1	1	0.417	1600
7	1650	1	1	0.794	1502	1	1	0.685	1520	1	1	0.472	1502
8	1600	1	1	0.813	1501	1	1	0.714	1550	1	1	0.52	1501
9	1700	1	1	0.828	1510	0	0	0.649	--	1	1	0.561	1520
10	1800	1	1	0.842	1520	1	1	0.677	1600	1	1	0.595	1550

Table 2: Simulation Result of Vm1 Vm2 and Vm3

The Running of different algorithms on each virtual machine provide whole system a different kind of robustness, if all virtual nodes running same algorithm and faults occur the whole system may get down. But if running algorithms are variant in nature chances of coincident failure never happen. We assume that each algorithm consist of many subtask and each subtask complete in one computing cycle.

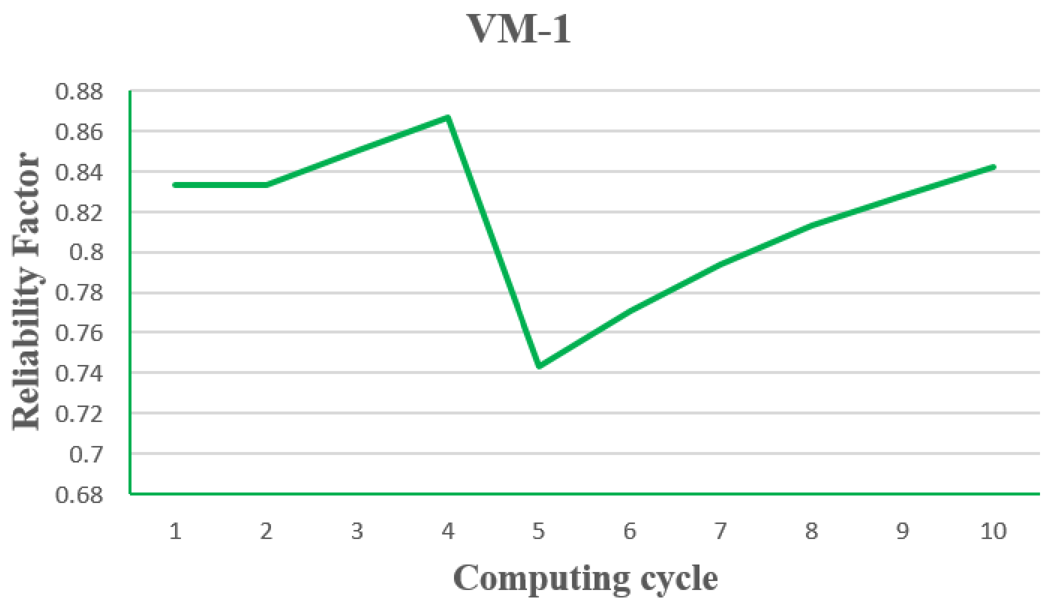

Fig 4. Virtual Machine -1

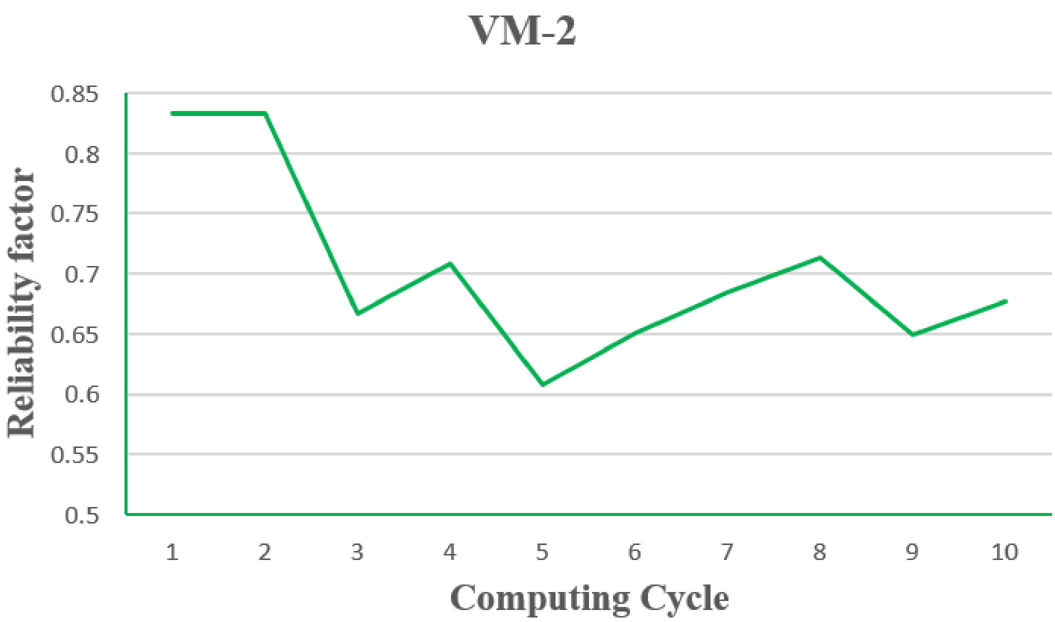

Fig 5. Virtual machine-2

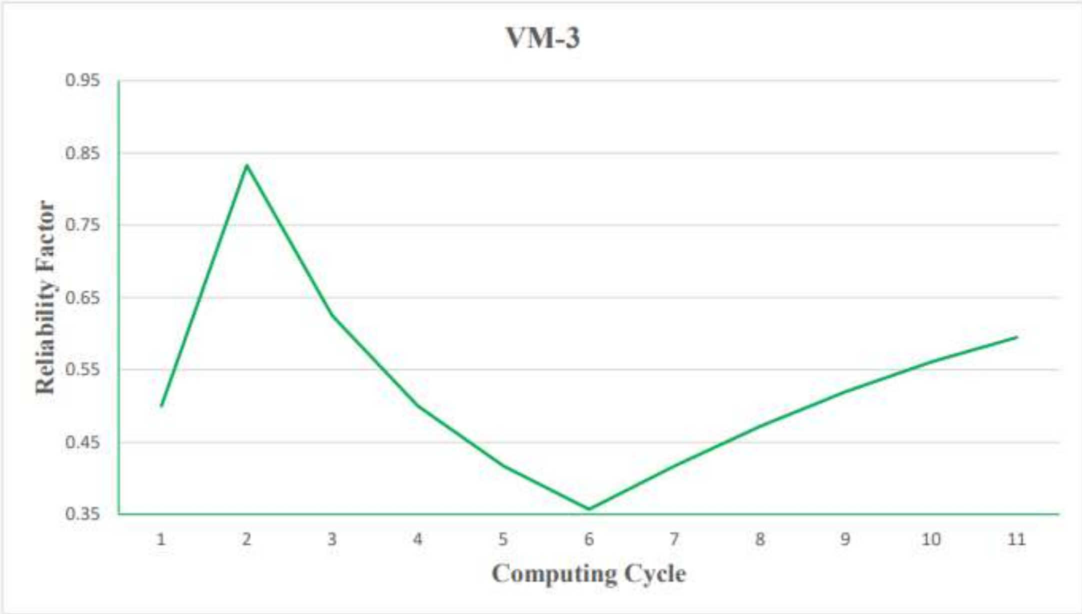

Fig 6. Virtual Machine-3

In the node selection mechanism (NSM) all the nodes who successfully execute the algorithm the submodule named status flag got updated by 1 (SF=1), after that job deadline check submodule checks how many tasks of algorithm completed within time, and get updated JDC=1 for successful cycle, and those who failed to complete the task within time, their corresponding JDC=0 and that node transfer to fault handler when migration of task perform. All those cycle completed in their deadline time, the reliability factor of the virtual machines improved and updated to cloud activity record table (CART). In Table 1. VM-2 cycle 9 shows that the reliability factor of the node still looking high but VM2 failed to complete the subtasks of algorithm this time, at this situation virtual machine not passed to final adjudication sub module, rather it will passed to fault handler and load balancer will inform not to assign task to this particular node. The final adjudication module having only those nodes whose reliability factors are high and their SF and JDC value is 1. Final adjudication module pass the node to service provider.

4.3 COMPARISON

Here we compare the simulation result of three virtual machines with existing models AFT and VFT [34] [35]. Figure 10, 11 and 12 clearly show that the proposed model produce high reliability factors which will provide more reliable node list that will assure the availability of resources and improve the throughput by selection of the highest reliable nodes.

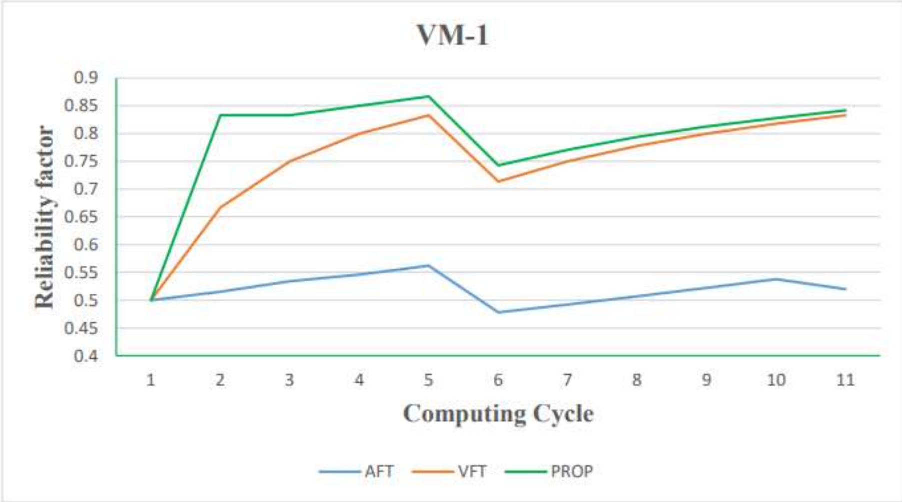

Fig 7. Comparison result of VM-1

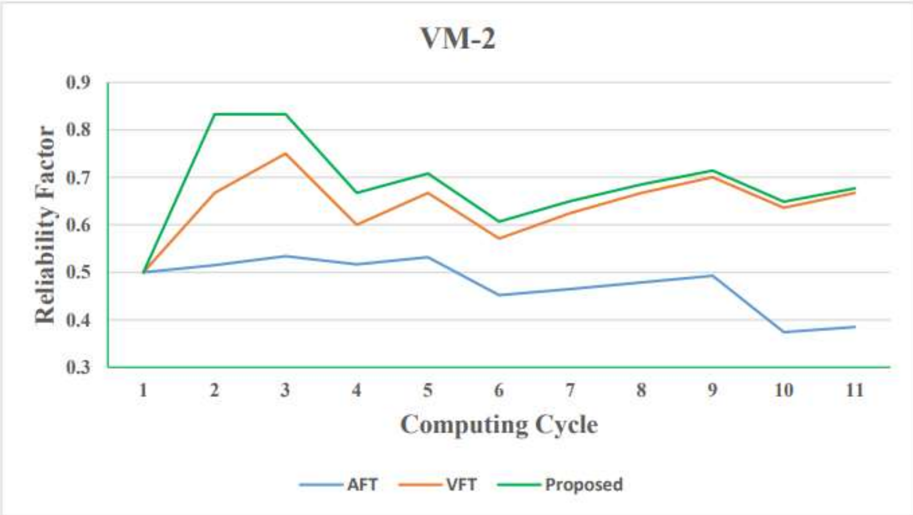

Fig 8. Comparison result of VM-2

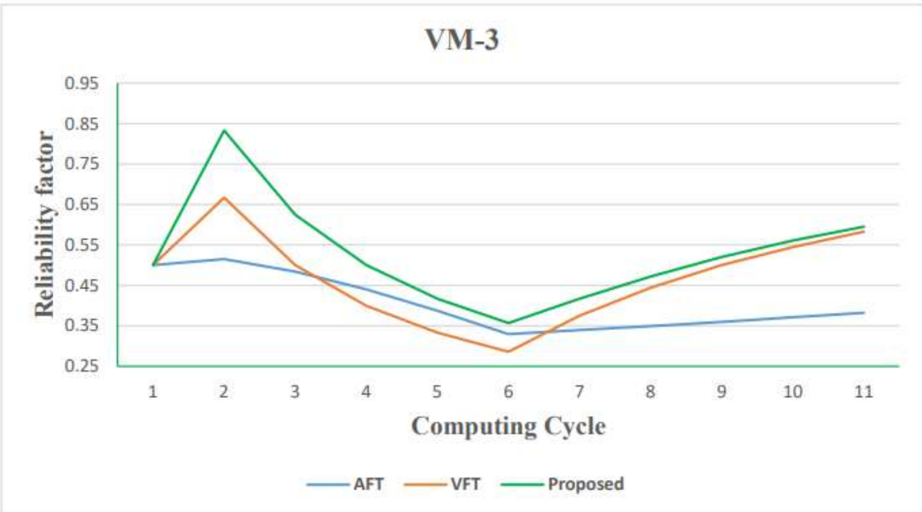

Fig 9. Comparison result of VM-3

The comparison of proposed model with AFT and VFT based on following parameters

1	**Standard Deviation**
2	**Mean rank**
3	**Reliability Factor R_f**

Table 2. Comparison parameters

4.3.1 Standard Deviation:

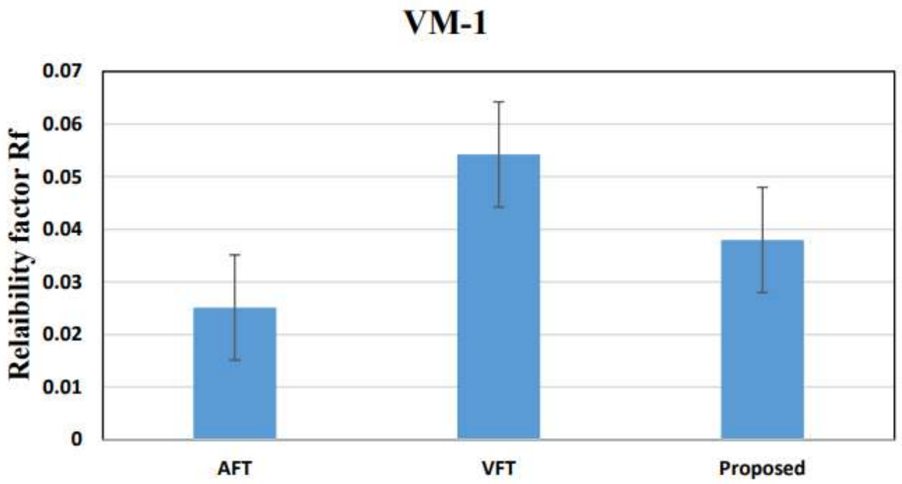

Fig 10. Comparison of Standard deviation

4.3.2 Mean Value Comparison:

Node	Mean Value		
	AFT	**VFT**	**Proposed**
VM-1	**0.5214**	**0.7743**	**0.8174**
VM-2	**0.4746**	**0.655**	**0.7015**
VM-3	**0.3958**	**0.4633**	**0.5297**

Table 3. Mean Values of Reliability Factors

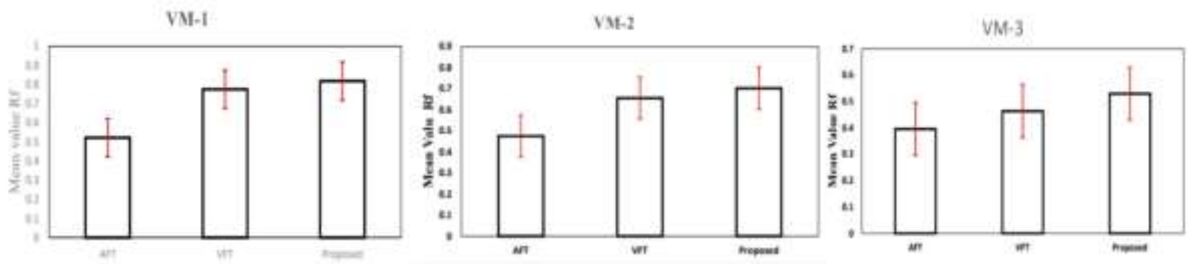

Fig 11. Comparison of Mean Value Reliability Factors

The mean Value reliability factor and standard deviation clearly indicating that, reliability factor of the nodes produced by proposed model (ITFTO) is far better than the existing model.

5. CONCLUSION

This paper presents an integrated ternate fault tolerant optimization in which the node having high reliability factor select for the job assignments. This method uses reverse cache recovery for creation of ruler node and during the migration, unnecessarily all node need not to create the check pointing. In most of the situation fault detected and handle by the controller unit run with the each node where FTdaemon act to predict the node going to fail and replace by the available list of high reliability factor nodes. After analyzing the section 4 it is reflected that the proposed model is highly resilience in nature to handle the fault, and mean value comparison shows that the reliability of the virtual nodes in proposed model are far high. In future we will work on the intelligent load balancer that will optimize the performance of virtual machines and use of machine learning to understand the key reason of the failure into a large scale very dynamic environment.

ACKNOLEDGEMENT: This work is acknowledged under Integral University manuscript No IU/R&D/2022-MCN 0001577.

REFERENCES

[1] K. Bilal, O. Khalid, S. Ur, R. Malik, M. Usman, and S. Khan, "Fault Tolerance in the Cloud," no. ITProPortal, pp. 1–13, 2012.

[2] BuyyaRajkumar, Ranjan Rajiv. Federated resource management in grid and cloud computing systems. JFuture GenerComputSyst2011;26(5):1189–91)

[3] Hughes, Kyle, Peter di Pasquale, Alessandra Babuscia, and Lorraine Fesq. "On-demand Command and Control of ASTERIA with Cloud-based Ground Station Services." In 2021 IEEE Aerospace Conference (50100), pp. 1-15. IEEE, 2021.

[4] Poulová, Petra, Blanka Klímová, and Martin Švarc. "Cloud Computing in the World and Czech Republic—A Comparative Study." In Advances in Computer, Communication and Computational Sciences, pp. 771-778. Springer, Singapore, 2021.

[5] Sangho Yi, Member, IEEE, Artur Andrzejak, and Derrick Kondo, Member, IEEE, "Monetary Cost-Aware Checkpointing and Migration on Amazon Cloud Spot Instances", IEEE Transactions on Services Computing, vol. x, no. x, month 201X.

[6] Mostafa Ghobaei-Arani Fault tolerance techniques in Cloudstorage International Journal of Database Theory and Application Vol.8, No.4 (2015)].

[7] ITProPortal, "ITProPortal.com: 24/7 Tech Commentary & Analysis," 2012. [Online]. Available: http://www.itproportal.com/. [Accessed: 24-Jun-2015].

[8] Pankaj Deep Kaur Kanu Priya Fault Tolerance Techniques and Architectures in Cloud Computing -A Comparative Analysis IEEE 2015

[9] Bianca Schroeder, Garth A. Gibson, "A large scale study of failures in high-performance-computing systems," In International Conference on Dependable Systems and Networks (DSN'06)

[10] A. Ganesh, M. Sandhya, and S. Shankar, "A study on fault tolerance methods in cloud computing," 2014 IEEE Int. Adv. Comput. Conf., pp. 844–849, 2014

[11] Karthikeyan, L., C. Vijayakumaran, S. Chitra, and Samydurai Arumugam. "Saldeft: Selfadaptive learning differential evolution based optimal physical machine selection for fault tolerance problem in cloud." Wireless Personal Communications 118, no. 2 (2021): 1453- 1480

[12] Hughes, Kyle, Peter di Pasquale, Alessandra Babuscia, and Lorraine Fesq. "On-demand Command and Control of ASTERIA with Cloud-based Ground Station Services." In 2021 IEEE Aerospace Conference (50100), pp. 1-15. IEEE, 2021

[13] Pinheiro Junior, Luiz, Maria Alexandra Cunha, Marijn Janssen, and Ricardo Matheus. "Towards a Framework for

Cloud Computing use by Governments: Leaders, Followers and Laggers." In The 21st Annual International Conference on Digital Government Research, pp. 155-163. 2020

[14] Benjamin Lussier, Alexandre Lampe, Raja Chatila,JerremieGuiochet, Felix Ingrand, Marc Olivier Killjian, David Powell Fault Tolerence in autonomous system:How and How much? LAAS-CNRS 7 Avenue du ColonrlRoche, F-31077Toulouse Cedex 04, France.

[15] D. Sun, G. Chang, C. Miao, and X. Wang, "Analyzing, modeling and evaluating dynamic adaptive fault tolerance strategies in cloud computing environments," Journal of Supercomputing, vol. 66, no. 1. J Suercomputer (), pp. 193–228, 2013

[16] M. Pradesh, "A survey on various fault tolerant approaches for cloud environment during load balancing," vol. 4, no. 6, pp. 25–34, 2014.

[17] Siva Sathya S, Syam Babu K. Survey of fault tolerant techniques for grid. Comput. Sci. Rev. 2010; 4(2): 101–120

[18] A. Tchana, L. Broto, and D. Hagimont, "Approaches to cloud computing fault tolerance," IEEE CITS 2012–2012 Int. Conf. Comput. Inf. Telecommun. Syst., 2012

[19] Kaur J, Kinger S. Efficient algorithm for fault tolerance in cloud computing. 2014 IJCSIT Int. J. Comput. Sci. Inf. Technol. 2014; 5: 6278–6281

[20] K. J. Naik and N. Satyanarayana, "A novel fault-tolerant task scheduling algorithm for computational grids," 2013 15th Int. Conf. Adv. Comput. Technol., pp. 1–6, 2013.

[21] Maloney A, Goscinski A. A survey and review of the current state of rollback-recovery for cluster systems. Concurr. Comput. Pract. Exp. 2009; 22(6): 685–701

[22] Kim HKH, Kang SKS, Yeom HY. Node selection for a fault-tolerant streaming service on a peer-to-peer network. 2003 Int. Conf. Multimed. Expo. ICME '03. Proc. (Cat. No.03TH8698) 2003; 2(1): 6–12.

[23] Ray, Benay, Avirup Saha, Sunirmal Khatua, and Sarbani Roy. "Proactive fault-tolerance technique to enhance reliability of cloud service in cloud federation environment." IEEE Transactions on Cloud Computing (2020)

[24] Qiang W, Jiang C, Ran L, Zou D, Services HJ. CDMCR: multi-level fault-tolerant system for distributed applications in cloud. International Journal of Applied Engineering Research 2015; 9(22): 5968–5974.

[25] Jung G, Joshi KR, Hiltunen MA, Schlichting RD, Pu C. Performance and availability aware regeneration for cloud based multitier applications. Proc. Int. Conf. Dependable Syst. Networks 2010: 497–506.

[26] Agbaria A, Friedman R. Virtual-machine-based heterogeneous checkpointing. Softw. - Pract. Exp. 2002; 32(12): 1175–1192.

[27] S. Shen, A. Iosup, A. Israel, W. Cirne, D. Raz, and D. Epema, "An availability-on-demand mechanism for datacenters," 2015 15th IEEE/ACM Int. Symp. Clust. Cloud Grid Comput., pp. 495–504, 2015.

[28] Chen G, Jin H, DeqingZou B, Zhou B, Qiang W. A lightweight software fault-tolerance system in the cloud enviroment. Concurr. Comput. Pract. Exp. 2015; 22(6): 685–701.

[29] Rong, and Haibo Chen. "Asymmetric virtual machine replication for low latency and high available service." Science China Information Sciences 61, no. 9 (2018): 1-15

[30] Zhou, Ao, Qibo Sun, and Jinglin Li. "Enhancing reliability via checkpointing in cloud computing systems." China Communications 14, no. 7 (2017): 1-10

[31] D. Caromel, C. Delbe, A. D. Costanzo, M. Leyton, ProActive: An Integrated Platform for Programming and Running Applications on Grids and P2P Systems, Computational Methods in Science and Technology 12(1), pp 69-77, 2006

[32] Bashir Mohammed1,†, Mariam Kiran2 ,Irfan-Ullah Awan3 and Kabiru .M. Maiyama4 2016 IEEE 4th International Conference on Future Internet of Things and Cloud ©2016 IEEE

[33] Ifeanyi P. Egwutuoha, Shiping Chen, David Levy, Bran Selic, "A Fault Tolerance Framework for High Performance Computing in Cloud", 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2012.

[34] Sheheryar Malik, Fabrice Huet, "Adaptive Fault Tolerance in Real Time Cloud Computing", IEEE World Congress on Services, 2011

[35] P. Das and P. M. Khilar, "VFT: A virtualization and fault tolerance approach for cloud computing," 2013 IEEE Conf. Inf. Commun. Technol. ICT 2013, no. Ict, pp. 473–478, 2013.