# Operative analysis of relevant factors affecting fault tolerance during load balancing using effective algorithms



**Bikash Chandra Pattanaik[1*], Bidush Kumar Sahoo[2], Bibudhendu Pati[3]**

***Corresponding Author:** Bikash Chandra Pattanaik
**Address:** *[1]Department of Computer Science and Engineering, Gandhi Institute for Education and Technology, Baniatangi, Khordha, Odisha, India, **Email:** bikashpatnaik73@gmail.com
Affiliated to Biju Patnaik University of Technology, Rourkela, Odisha, India
[2]Department of Computer Science and Engineering, GIET University, Gunupur, Odisha, India, **Email:** bidush.sahoo@gmail.com
[3]Department of Computer Science, Ramadevi Women's University, Bhubaneswar, Odisha, India,
[3]**Email:** patibibudhendu@gmail.com

## Abstract

In the current era, Cloud Computing is becoming an efficient solution for significant issues in virtual resource sharing. Because of its solution providing capability in various sectors, the Cloud Computing users are growing significantly. So, in order to improve the performance of cloud servers, load balancing approaches plays a vital role. Considering the fault tolerance during load balancing in cloud computing environment, we have considered some vital factors that may affect the fault tolerance in load balancing. Using various algorithms, we have analyzed the impact of factors affecting fault tolerance during load balancing in cloud computing.

**Keywords:** Cloud computing, load balancing, fault tolerance, efficient

**DOI Number: 10.14704/nq.2022.20.9.NQ44196** **Neuro Quantology 2022; 20(9):1702-1709**

## 1. Introduction

Cloud computing is a technology that provides computing resources over the Internet using numerous samples and layers of abstraction. The beginning of cloud computing has been called the single most influential change in information technology since the invention of the Internet. This change has enabled many people to take action, including individuals, the community, and large corporations. Cloud computing is the delivery of applications as services to the mass, varying from end users hosting their documents on the Internet to enterprises outsourcing their complete IT infrastructure. Although cloud computing has been adopted extensively by the industry, many research issues remain to be fully addressed, such as fault tolerance, workflow scheduling, workflow management, security, and privacy. Fault tolerance is a key issue in software design. It concerns techniques that enable a system to remain functional after errors have been introduced during development. Fault tolerance approaches help to prevent a system from failure in the event of a fault. Fault tolerance is a key advantage of cloud computing, providing failure recovery, lower costs, and improved performance metrics.

### 1.1 Fault tolerance and Load balancing

#### 1.1.1 Fault Tolerance

The fault-tolerant design strategy enables a system to operate despite failures—such as memory, disk space, network grid, and processor (CPU) problems—by spotting and circumventing them. The fault tolerance methodology's purpose is to keep the system up and running with high system stability and reliability. Cloud computing providers aim to improve their platform by applying a fault tolerance strategy to achieve a high availability system based on the significance of fault tolerance in cloud computing systems. The letdowns in cloud computing can be divided into two categories. However, first software susceptibility includes abuse of information and incomplete data from the source and failure of hardware like defective or lagging VMs and the exception of storage access.

### 1.1.2 Fault tolerance policies

#### *1.1.2.1 Reactive Fault Tolerance*:

When a cloud system develops faults, it is possible to tolerate the faulty behavior using reactive tolerance. A reactive policy can be designed to respond to new failures and isolate failures in the cloud system.

#### *1.1.2.2 Proactive Fault Tolerance*:

The theory of proactive fault avoidance is much more advanced than the theory of reactive fault recovery. The latter approach concerns itself with identifying faults as they occur and aiming to avoid both their occurrence and the impact of faulty operations on the affected system.

### 1.2 Load Balancing

Load balancing is the process by which a system distributes or redistributes its load among multiple resources. It can improve system performance by distributing workloads over numerous computing resources in cloud computing environments. Load balancing uses techniques such as migrating virtual machines to other compute nodes to distribute workloads across multiple resources [3]. Figure 1 illustrates the load balancing model in cloud computing. The load balancer receives incoming requests from users and forwards them to the load balancing algorithm, which distributes incoming requests among virtual machines (VMs) according to rules set by administrators. The foremost objective of the load balancer is to determine which user role should be delegated to a specific VM. Load balancing algorithms are based on two factors: static algorithms and dynamic algorithms.

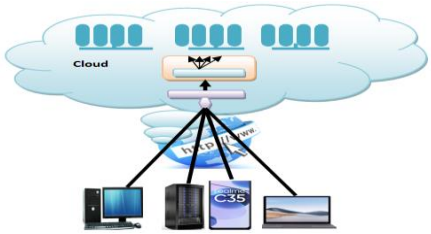


**Figure 1:** Load Balancing in Cloud Computing

### 1.2.1 Static Load balancing

These algorithms depend on previous environmental knowledge, such as requirements for connected nodes and memory. The load-balancing decision is taken in these algorithms during the compilation process. These algorithms divide the load equally between nodes according to previous system knowledge. Therefore, few load variations exist, and process runtime adjustments are not considered [1, 2].

### 1.2.2 Dynamic Load Balancing

Dynamic algorithm decisions are based on the current state of a system, which changes at runtime and is continually monitored, so that previous information about that system is not taken into account. Many heterogeneous systems use algorithms for dynamic load balancing.



## 2. Related Work

The popularity of cloud computing has led users to demand fault tolerance. However, several factors contribute to the instability of cloud computing services, including network connectivity and virtual machine management issues. Much research has therefore been applied to cloud computing fault tolerance. Huf et al. [7] proposed, evaluated, and implemented fault-tolerant techniques for xApps in the RIC platform. These techniques were made available to xApp writers as a library called RFT (RIC Fault Tolerance). Zeng et al. [8] present a hardware architecture for stateful layer-4 LBs called Tiara that aims to support a high traffic rate (1 Tbps), a large number of concurrent flows (> 10M), and many new connections per second (1M), without any assumption on traffic patterns. Irian et al. [9] studied how to scale and balance transparent virtual network functions (VN-Fs). The authors analyze different aspects of this problem, such as loop avoidance when scaling-out actions and bidirectional flow affinity. To address this problem, SDN controllers and Open Flow switches were implemented. A scientific workflow is a sequence of computations or data manipulations that represents the process of a scientific research program or experiment [10]. A combination of resource-intensive computing and abundant storage is required to implement a scientific workflow application. But conventional systems may not provide the necessary storage and computing power. Cloud computing is a good choice for scientific workflow systems because the infinite capacity and massive scalability eliminate barriers to the number of experiments that labs can

conduct. When costs are bundled into a flat monthly fee, individual researchers and large institutions can save money [11]. Alaei et al. introduced a mechanism for handling system faults in real time. This approach creates the determination that keeps the adaptive reliability of processing nodes. The reliability of the virtual machine changes every computing cycle because of the adaptive factor, which relates to the correctness of results [12]. Zhao et al. proposed a Low Latency Fault Tolerance System (LLFT) for cloud distributed applications that are used to tolerate faults in distributed systems. This model is based on the replication method; it replicates applications using various replication processes to save them from defects [13, 14]. Ragmani et al. proposed a reactive fault tolerance model called Fault Tolerance Management (FTM) that exploits checkpoint and replication policies to provide cloud-on-demand service [15, 16]. Suresh et al. [17] presented a proactive fault tolerance utilizing Naïve Bayes machine learning. They generated a list of the most frequently faulted nodes and then applied techniques to assure enhanced reliability for these nodes. Some of the works are summarized in table 1

**Table 1:** Summary of Fault tolerance techniques in cloud computing

| Author and Year | Methods | Findings |
|---|---|---|
| Y. Zhu et al. (2020)[18] | FT-PBLAS(Fault tolerance parallel basic linear algebra subprograms) | The author proposes FT-PBLAS, a fault-tolerant version of the parallel basic linear algebra subprograms (PBLAS). It combines the features of PBLAS with those of fault-tolerant libraries. Failure proposes a block-checksum approach for non-fatal errors and a scheme for addressing nodes. |
| Orogun A.O et al. (2020) [19] | Reliability and Replication Techniques | This paper will discuss the principles of system reliability and replication techniques that address fault-tolerance requirements. The author proposed a new scheme called primary-peer replication (PPR) to overcome the limitations of the coordinator-cohort and leader-follower schemes. |
| Har simratsingh et al. (2020) [20] | Replication Technique | A fault-tolerant workflow scheduling approach is described, which uses replication heuristics and checks in the cloud. The proposed technique improves upon metrics like wastage of resources and Resource usage while keeping an acceptable increase in Makespan compared to the vanilla Heterogeneous Earliest Finish Time. |
| Archana Pandita et al. (2018) [21] | Rescue Workflow | A comparative analysis of fault-tolerance algorithms in cloud computing. The authors aim to study and compare various scheduling algorithms for cloud computing, focusing on fault tolerance. They also briefly discuss the advantages and disadvantages of these algorithms. |
| Bashir Mohammed et al. (2019) [22] | Prediction | The authors used several machine learning algorithms to improve the predictive accuracy of their failure prediction model. They used their model to predict failures in producing a medical implant and compared that prediction with actual results. |
| Kuan Wang et al. (2021) [23] | Fault Induction | Energy function virtualization with game theory for fault-tolerant smart grid. Simulation results indicate that the proposed technique can dynamically adjust the delay factor to shorten emergency response time. |
| Chindamani M. et al.(2021) [24] | Performance analysis | This paper presents performance analyses of fault-tolerance techniques for cascaded multilevel inverters. |
| Dongqiu Xing et al. (2021) [25] | Machine learning | This study introduces a multi-source fault identification method, which uses a combined deep learning strategy to identify multi-source faults in complex industrial systems. The framework consists of quality extraction and classifier design. |
| AnatoliyGorbenk et al. (2020) [26] | Replication Technique | The authors examine the tradeoffs between fault tolerance and performance in fault-tolerant distributed systems and replicated databases built over the Internet. This paper may interest |

| | | |
|---|---|---|
| | | software engineers, system architects, and researchers who develop Internet-scale distributed computer systems and cloud solutions. |
| Nalini Niranjan et al. (2018) [27] | N-Version and recovery block | Cloud computing uses software configurations stored in cloud storage to make fault-tolerant systems. Software reliability and fault tolerance have been achieved cost-effectively by employing these software repair schemes. |

## 3. Background Study

## Influential Metrics

### A. Throughput

Throughput shows a VM's number of user demands (assignments) performed per unit time. The throughput value defines the system performance. Increased throughput indicates that the system performance is good. The throughput of the system is inversely proportional to the makespan of the system.

### B. Reliability

Reliability will invariably perform according to the system specifications. During task execution, if a failure occurs, the task is transferred to other resources (VMs) to improve the system's reliability. A reliable system enhances the stability of the system.

### C. Accuracy

It determines the perfection of task execution results. Accuracy is a measurement's ability that matches the actual value of the measured task execution. Currently, IT industries give more importance to system accuracy according to user demand. The accuracy value slightly degrades the system makespan.

### D. Resource Utilization

The use of resources in a data center is important to ensure that the electricity and other resources are used efficiently. The Cloud's use of resources can be optimized through various techniques.

### E. Scalability

A system or model feature defines the technique's ability to perform under unexpected circumstances. It means the survival level for a balanced approach when the number or size of jobs or workload is raised. In a scalable cloud system, resources will be rescaled periodically.

### F. Response Time

Respond to any task in the time required. In other words, it is the sum of communication, waiting, and service time. Thus, the system performance is inversely proportional to the response time. The optimal response time outcomes in a more suitable makespan value.



### G. Performance

The attributes normally check the efficiency of the prepared system. The performance of any procedure will be foremost when the reply time is minimum and throughput is increased.

### H. Cost effectiveness

The cost difference between on-demand and reserved instances depends on the percentage of resource utilization. For example, 50% resource utilization reduces overall costs by 49% while fully utilized. Cloud users try to depreciate the cost of resource provisioning by degrading the on-demand resource cost and over-subscribed resource cost of over-provisioning and under-provisioning (Shahidinejad et al., 2021) [6].

### I. Usability

It signifies when a consumer can utilize a method for achieving its necessity with the constants of efficiency and satisfaction [4, 5].

### J.Availability

Availability of a system is proportional to the reliability of the system. Basically it is the probability of an item with functions at the given period of time.

### K. Overhead Associated

The overhead are necessary for tax improvement and evolution of the inter and intra processor communication. It always should be minimized.

### L. QoS Satisfaction

Quality of service (QoS) is the measurement of a service's performance, including its availability, reliability and efficiency. It allows organizations to prioritize specific high-performance applications.

## M. Failure Rate

The main goal of this study is to enhance the system reliability and thereby reduce failure occurrences, which directly affects overall system reliability.

## N. Load balancing

Load balancing is a system administration technique that allows the distribution of a computing workload across multiple hardware or virtual machine instances. This helps to improve overall system performance in cloud environments and can be achieved by using different types of load balancers such as software, hardware, or network devices.

## 4. Experimentation and Result analysis

We have considered various factors like scalability, cost-effectiveness, response time, QoS satisfaction, throughput, and the overhead associated with resource utilization, availability, usability, and reliability for fault tolerance during load balancing in cloud computing environment. We have analyzed and validated using SPSS Statistical tool. The effecting percentage of those factors on fault tolerance is shown in figure 2.We experimented with various load-balancing algorithms for our work. Equation 1 shows the generalized linear regression equation. We implemented FTSM, Load balanced user demand aware scheduling Algorithm and a Multi-constrained load balancing algorithm in figure 3. We performed experiments with the loan balancing algorithm and obtained the results by tuning various hyper-parameters for each of the algorithms.

In this experiment, we have tuned each algorithm with parameters like scalability, cost-effectiveness, response time, QoS satisfaction, throughput, and the overhead associated with resource utilization, availability, usability, and reliability. Based on the values taken for these parameters and performance by the load balancing algorithm, we perform a linear regression analysis to determine which parameter is affecting the fault tolerance load balancing algorithm.

For FTSM, parameters like scalability, performance, reliability, the overhead associated, usability, and availability impact the performance of the fault tolerance algorithm more as compared to other parameters or features. The reliability of the FTSM algorithm is greater, but it is not a cost-effective approach for achieving the aim. But it efficiently does the resource utilization, and the usability also impacts the performance, like the algorithm's accuracy.

However, load balancing is affected by reliability, scalability, overhead association, usability, and availability.

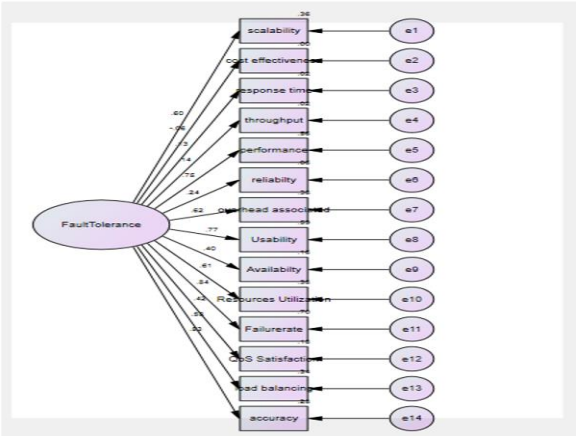


**Figure 2:** Validation of the factors affecting fault tolerance during load balancing

For a Multi-constrained load balancing algorithm, parameters like cost effectiveness, throughput, performance, failure rate, usability, overhead associated, resource utilization and availability impact the performance of the fault tolerance algorithm more as compared to other parameters or features. The usability and availability of the Multi-constrained algorithm are more, but it is not a scalable approach for achieving the aim in figure 4.

However, load balancing is affected by throughput, overhead associated, usability, availability, resource utilization, failure rate, QoS Satisfaction, and scalability.

$$Y = m * x + c \qquad (1)$$

### *4.1 Linear Regression Analysis for FTSM algorithm*

Accuracy = 0.05003243 * scalability + 0.05456523 * cost effectiveness + (-0.1173057) * response time + (-1.00892429) * throughput + (-0.09557523)* performance + 0.70593843 * reliability + 0.20119831 * overhead associated + 0.66371231 * usability + 0.03548487 * availability + 0.26582946 * resource utilization + (-0.25806423) * failure rate + (-0.61568341) * QoS Satisfaction + 3.49943692717626

RMSE (Accuracy) = 1.6499744680105275

R2 Score = -0.21536417191367074
Mean Squared Error = 2.722415745086623
Mean absolute error = 1.54424626666636

Load balancing = 0.00496797 * scalability + (-0.04215542)*throughput + (-0.79586215) * performance + 0.2672997 * reliability + 0.74379609 *overhead associated + 0.61038481 * usability + (-0.02586913) * availability + 0.17761167 * resource utilization + 0.34381866 failure rate + (-0.44219805) * QoS Satisfaction + (-0.06203606) * cost effectiveness + 0.35386568 * response time + (-1.115194401979533)

Mean absolute error = 1.54424626666636
r2 score = -0.21536417191367074
Mean Squared Error = 2.722415745086623
RMSE = 1.6499744680105275

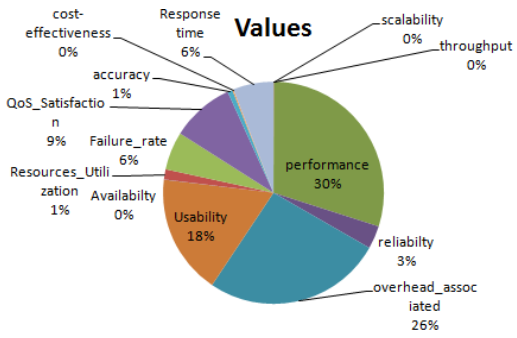


**Figure 3:** Parameters affecting FTSM

## *4.2 Linear Regression Analysis for Multi-constrained load balancing algorithm*

Accuracy = (-0.45081145)* scalability + 0.28301795* cost + (-0.41518376)* time + 0.34930394 * throughput + 0.66492922 * performance + (-0.44146685)* reliability + (-1.27350941)* overhead associated + 0.0851996 + 0.88529215 * Usability + 0.54767592 * Availability + 0.32085981 * Resources Utilization + 0.14173288 * Failure rate + 0.59589471 * QoS_Satisfaction + (-0.9591756474271858)

Mean absolute error = 2.5149417464484936
r2 score = -4.568698097500735
Mean Squared Error = 12.473883738401648
RMSE = 3.5318385776252073

Load Balancing= 0.66492922 * throughput + (-0.44146685) *performance + (-1.27350941) * reliability + 0.0851996 * overhead associated + 0.88529215 * Usability + 0.54767592 * Availability + 0.32085981* Resources Utilization + 0.14173288* Failure rate + 0.59589471* QoS_Satisfaction + 0.28301795 *scalability + (-0.41518376)* Cost effectiveness + 0.34930394 * response Time + (-0.9591756474271858)

Mean absolute error = 2.5149417464484936
r2 score = -4.568698097500735
Mean Squared Error = 12.473883738401648
RMSE = 3.5318385776252073

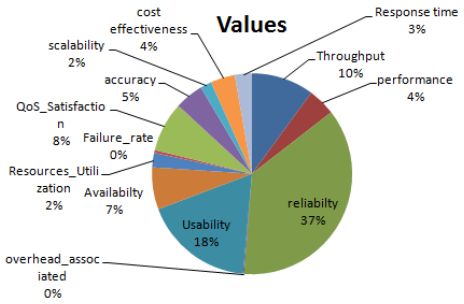


**Figure 4:** Factors affecting multi constrained load balancing algorithm.

## *4.3 Linear Regression Analysis for Load balanced user demand aware scheduling Algorithm*

Accuracy= 0.35420374 * scalability + (-0.9126594) * cost effectiveness + 0.00841523 * response time + (-0.4307548) * throughput + (-0.37852331) * performance + 0.48781135 * reliability + 0.47431815 * overhead associated + 0.40485273* Usability + 1.28238836* Availability + 0.0167595 * Resources Utilization + (-0.65915409) * Failure rate + 0.03433198* QoS Satisfaction + 6.321096535431893

Mean absolute error = 3.3800141169628555
r2 score = -5.55108892947531
Mean Squared Error = 20.963484574320994
RMSE = 4.578589801928208

Load Balancing = (-0.24823859)* throughput + (-0.31371659)* performance + 0.74964751* reliability + 0.61717884 * overhead associated+ (-0.10138049) * Usability + 0.18884174* Resources Utilization + (-0.10118584)* Failure rate +1.01178312 * QoS_Satisfaction + 0.2924768 *scalability + (-0.11455169) * Cost effectiveness + 0.3051568* response time + 6.321096535431893.

Mean absolute error = 3.3800141169628555
r2 score = -5.55108892947531
Mean Squared Error = 20.963484574320994

RMSE = 4.578589801928208

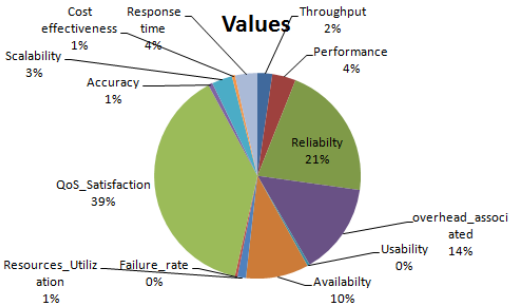


**Figure 5:** Factors affecting load balanced user demand algorithm.

For load balanced user demand aware scheduling Algorithm, parameters like scalability, response time, reliability, overhead association, usability, availability, resource utilization, QoS Satisfaction impact the performance of the fault tolerance algorithm more as compared to other parameters or features shown in figure 5. The QoS Satisfaction and availability of the user demand aware scheduling algorithm are more, but it is not a cost effectiveness approach for achieving the aim.

However, load balancing is affected by reliability, overhead associated, resource utilization, QoS Satisfaction, scalability and response time.

We also compared the results of r2 score of each of the algorithm for fault tolerance in figure 2.

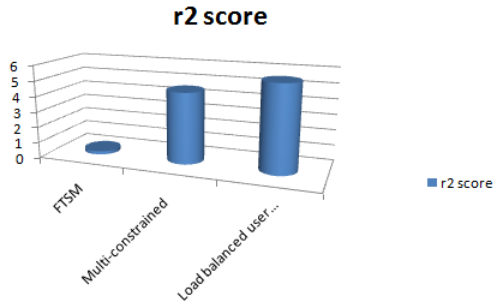


**Figure 2:** Comparison between the algorithms

The r2 score of the Load balanced user demand aware scheduling Algorithm is more compared to remaining two algorithms, if we consider all the parameters like scalability, cost effectiveness, response time, throughput, performance, reliability, overhead associated, usability, availability, resource utilization, failure rate, QoS satisfaction and load balancing. Hence, we can consider Load balanced user demand aware scheduling algorithm as a better approach for fault tolerance when we consider the above-mentioned parameters.

## 5. Conclusion and Future Work

Cloud computing environment necessitates proper management of load balancing in order for efficient usage of the cloud resources. There are various factors that may significantly affect the procedure of load balancing in a cloud computing environment. In this work, we have considered the prime factors affecting the fault tolerance during load balancing in cloud computing environment. We have utilized different algorithms for verifying the effectiveness of the significant factors like scalability, response time, reliability, overhead association, usability, availability, resource utilization etc. Out of the validation process we found failure rate has the highest impact on fault tolerance during load balancing in cloud computing environment. We have also analyzed the load balancing using efficient algorithms and found demand aware scheduling algorithm as a better approach for fault tolerance. As an extension of the current research work, scalability analysis can improve the efficiency of a cloud computing environment so far as load balancing is concerned.



## References

Hota, N., & Pattanayak, B. K. (2021). Cloud computing load balancing using Amazon web service technology. In *Progress in Advanced Computing and Intelligent Engineering* (pp. 661-669). Springer, Singapore.

Pati, A., Parhi, M., & Pattanayak, B. K. (2022). HeartFog: Fog Computing Enabled Ensemble Deep Learning Framework for Automatic Heart Disease Diagnosis. In *Intelligent and Cloud Computing* (pp. 39-53). Springer, Singapore.

Laha, S. R., Parhi, M., Pattnaik, S., Pattanayak, B. K., & Patnaik, S. (2020, October). Issues, Challenges and Techniques for Resource Provisioning in Computing Environment. In *2020 2nd International Conference on Applied Machine Learning (ICAML)* (pp. 157-161). IEEE.

Peng, S. L., Suseendran, G., & Balaganesh, D. (2020). *Intelligent Computing and Innovation on Data Science*. Springer Singapore.

Mohammadian, V., Navimipour, N. J., Hosseinzadeh, M., & Darwesh, A. (2020). Comprehensive and systematic study on the fault tolerance architectures in cloud computing. *Journal of Circuits, Systems and Computers*, *29*(15), 2050240.

Shahidinejad, A., Ghobaei-Arani, M., & Masdari, M. (2021). Resource provisioning using workload clustering in cloud computing environment: a hybrid approach. *Cluster Computing*, *24*(1), 319-342.

Huff, A., Hiltunen, M., & Duarte, E. P. (2021, May). RFT: Scalable and fault-tolerant microservices for the O-RAN control plane. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)* (pp. 402-409). IEEE.

Zeng, C., Luo, L., Zhang, T., Wang, Z., Li, L., Han, W., & Guo, C. (2022). Tiara: A scalable and efficient hardware acceleration architecture for stateful layer-4 load balancing. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)* (pp. 1345-1358).

Llorens-Carrodeguas, A., Leyva-Pupo, I., Cervelló-Pastor, C., Piñeiro, L., & Siddiqui, S. (2021). An SDN-Based Solution for Horizontal Auto-Scaling and Load Balancing of Transparent VNF Clusters. *Sensors*, *21*(24), 8283.

Naik, B. B., Singh, D., & Samaddar, A. B. (2021). Multi-objective virtual machine selection in cloud data centers using optimized scheduling. *Wireless Personal Communications*, *116*(3), 2501-2524.

Wang, Z. (2021). Research on Digital Economy and Human Resources Based on Fuzzy Clustering and Edge Computing. *Security and Communication Networks*, *2021*.

Alaei, M., Khorsand, R., &Ramezanpour, M. (2021). An adaptive fault detector strategy for scientific workflow scheduling based on improved differential evolution algorithm in cloud. *Applied Soft Computing*, *99*, 106895.

Cao, W., Liu, Z., Wang, P., Chen, S., Zhu, C., Zheng, S., & Ma, G. (2018). PolarFS: an ultra-low latency and failure resilient distributed file system for shared storage cloud database. *Proceedings of the VLDB Endowment*, *11*(12), 1849-1862.

Gencer, C., Topolnik, M., Ďurina, V., Demirci, E., Kahveci, E. B., Lukáš, A. G. O., & Katsifodimos, A. (2021). Hazelcast Jet: Low-latency stream processing at the 99.99 th percentile. *arXiv preprint arXiv:2103.10169*.

Ragmani, A., Elomri, A., Abghour, N., Moussaid, K., Rida, M., &Badidi, E. (2020). Adaptive fault-tolerant model for improving cloud computing performance using artificial neural network. *Procedia computer science*, *170*, 929-934.

Anuradha, S., & Kanmani, P. A Systematic Survey of Fault Tolerance in Cloud Environment.

Sathiyamoorthi, V., Keerthika, P., Suresh, P., Zhang, Z. J., Rao, A. P., & Logeswaran, K. (2021). Adaptive fault tolerant resource allocation scheme for cloud computing environments. *Journal of Organizational and End User Computing (JOEUC)*, *33*(5), 135-152.

Zhu, Y., Liu, Y., & Zhang, G. (2020). FT-PBLAS: PBLAS-based fault-tolerant linear algebra computation on high-performance computing systems. *IEEE Access*, *8*, 42674-42688.

Orogun, A. O. (2020)Reliability And Replication Techniques For Improved Fault Tolerance In Distributed Systems.

Setlur, A. R., Nirmala, S. J., Singh, H. S., & Khoriya, S. (2020). An efficient fault tolerant workflow scheduling approach using replication heuristics and checkpointing in the cloud. *Journal of Parallel and Distributed Computing*, *136*, 14-28.

Pandita, A., Upadhyay, P. K., & Joshi, N. (2018, December). Fault tolerance based comparative analysis of scheduling algorithms in cloud computing. In *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)* (pp. 1-6). IEEE.

Mohammed, B., Awan, I., Ugail, H., & Younas, M. (2019). Failure prediction using machine learning in a virtualized HPC system and application. *Cluster Computing*, *22*(2), 471-485.

Zhang, X., Xu, G., Kuang, J., Suo, L., Zhang, S., &Khalique, U. (2021). A Three-Phase Current Tacholess Envelope Order Analysis Method for Feature Extraction of Planetary Gearbox under Variable Speed Conditions. *Sensors*, *21*(17), 5714.

Xing, D., Chen, R., Qi, L., Zhao, J., & Wang, Y. (2020). Multi-source fault identification based on combined deep learning. In *MATEC Web of Conferences* (Vol. 309, p. 03037). EDP Sciences.

Meyyappan, C., & Ravichandran, C. S. (2021, January). Performance analysis of fault-tolerance techniques towards solar fed cascaded multilevel inverter. In *2021 International Conference on Computer Communication and Informatics (ICCCI)* (pp. 1-7). IEEE.

Gorbenko, A., Karpenko, A., &Tarasyuk, O. (2020, May). Analysis of trade-offs in fault-tolerant distributed computing and replicated databases. In *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)* (pp. 1-6). IEEE.

Chinnaiah, M. R., & Niranjan, N. (2018). Fault tolerant software systems using software configurations for cloud computing. *Journal of Cloud Computing*, *7*(1), 1-17.