Academic Journal of Nawroz University (AJNU), Vol.13, No.2, 2024
This is an open access article distributed under the Creative Commons Attribution License
Copyright ©2017. e-ISSN: 2520-789X
https://doi.org/10.25007/ajnu.v13n1a2012

Leveraging Distributed Systems for Fault-Tolerant Cloud Computing: A Review of Strategies and Frameworks

Saman M. Almufti[1,2] and Subhi R. M. Zeebaree[3]

[1] IT Dept., Technical College of Informatics, Akre University for Applied Sciences, Duhok, Iraq.
[2] College of Science, Department of Computer Science, Nawroz University, Duhok, Kurdistan Region, Iraq.
[3] Energy Eng. Dept., Technical College of Engineering, Duhok Polytechnic University, Duhok, Iraq.

Abstract: Ensuring system availability and reliability is crucial in the quickly developing field of cloud computing. The importance of fault tolerance in cloud infrastructure systems grows as organizations become more reliant on it to support their critical operations. The purpose of this article is to investigate the intricate realm of cloud computing and distributed systems. Specifically, the paper will investigate the numerous forms of cloud computing, fault tolerance methods, and frameworks that enable cloud services to be robust and durable.
Cloud computing has transformed the way in which organizations and individuals access and administer computing resources. The paper discusses several deployment options, including public, private, hybrid, and multi-cloud environments, which provide organizations with the advantages of flexibility, scalability, and cost-effectiveness. The inherent flexibility of cloud computing renders it well-suited for a diverse range of applications, spanning from the hosting of websites to the execution of intricate data analytics processes.

Generally, cloud computing encounters substantial obstacles, including the need of maintaining uninterrupted service in the face of hardware failures, network outages, or software errors, despite its tremendous benefits. The critical importance of fault tolerance in this particular situation cannot be overstated, as it plays a pivotal role in maintaining the dependability and availability of the system.
The primary objective of this study is to examine the utilization of distributed systems as a means to augment fault tolerance within the realm of cloud computing and distributed systems. Distributed systems offer an optimal approach for addressing difficulties related to fault tolerance, owing to its intrinsic capability to divide workloads and data over several nodes. This approach utilizes redundancy, replication, and the ability to recover seamlessly from disturbances, hence enhancing the resilience and resource efficiency of cloud services. This research reviews novel techniques and frameworks that utilize distributed systems to create fault-tolerant cloud computing architectures, emphasizing their substantial influence on the cloud computing domain. In conclusion, this research report includes a comparative analysis table that encompasses twenty preceding works.
Keywords: IoT, Cloud computing, Distributed system, Fault-Tolerant Cloud Computing

1. Introductions:

In the changing environment of cloud computing, assuring service stability and availability has become critical. The importance of fault tolerance in cloud computing systems is growing and more enterprises depend on cloud infrastructure for critical tasks. This study reviews the complex realm of cloud computing and distributed systems, investigating various kinds, fault tolerance techniques, and frameworks that facilitate the implementation of reliable and resilient cloud services.

Cloud computing has transformed how organizations and individuals use and maintain their computers. The platform provides a wide array of deployment types, encompassing public, private, hybrid, and multi-cloud environments. Cloud computing has gained significant appeal due to its potential to offer flexibility, scalability, and cost-efficiency across a wide range of applications, including website hosting and complicated data analytics pipelines.

Nevertheless, the benefits of cloud computing are accompanied with a series of challenges, mostly centered around the need of maintaining uninterrupted service provision, even in the presence of hardware malfunctions, network disruptions, or software inconsistencies. Fault tolerance plays an essential role in ensuring the dependability and availability of a system, hence becoming a vital component in this context.

The use of distributed systems in cloud computing represents a notable paradigm change to enhancing fault tolerance and dependability in cloud-based ecosystems. Distributed systems are well-suited for tackling the difficulties related to fault tolerance due to their intrinsic ability to divide workloads and data over different nodes or resources. This methodology is predicated around the utilization of redundancy, replication, and the capacity to effortlessly restore operations following interruptions. The enhancement of cloud services is not only achieved, but it also results in the optimization of resource consumption and scalability. Through the utilization of distributed systems, cloud providers and organizations may ensure the continuous accessibility and functionality of their critical applications and data, even in the presence of faults or failures in individual components or nodes. This article aims to explore in-depth the novel methodologies and frameworks that utilize distributed systems for the development of fault-tolerant cloud computing architectures. It will demonstrate their significant impact on the whole landscape of cloud computing, highlighting its transformational nature.

The field of fault tolerance strategies in cloud computing covers a wide range of approaches, including but not limited to redundancy, replication, load balancing, and gentle degradation. The use of these measures is crucial in the effort to reduce interruptions and minimize the consequences of failures on cloud-based applications and services. A fault-tolerant system that exhibits robustness not only serves to enhance the user experience, but also provides protection against potential data loss and financial losses that may arise due to periods of system unavailability.

In order to attain fault tolerance in the realm of cloud computing, a multitude of frameworks and tools have surfaced, each meticulously crafted to tackle distinct issues and meet unique criteria. The objective of this study is to present a thorough examination of these frameworks, explaining their structures, functionalities, and practical implementations. This study investigates the role of distributed systems, including container orchestration platforms like Kubernetes, fault-tolerant storage solutions like Ceph, and serverless computing frameworks like AWS Lambda, in enhancing the fault tolerance of cloud computing environments.

In the ever-changing environment of cloud computing, assuring service stability and availability has become critical. The importance of fault tolerance in cloud computing systems is growing as more and more enterprises depend on cloud infrastructure for critical tasks. This study examines the complex realm of cloud computing and distributed systems, investigating various kinds, fault tolerance techniques, and frameworks that facilitate the implementation of reliable and resilient cloud services.

Cloud computing has transformed how organizations and people use and maintain their computers. The platform provides a wide array of deployment types, encompassing public, private, hybrid, and multi-cloud environments. Cloud computing has gained significant appeal due to its potential to offer flexibility, scalability, and cost-efficiency across a wide range of applications, including website hosting and complicated data analytics pipelines.

Nevertheless, the benefits of cloud computing are accompanied with a series of challenges, mostly centered around the need of maintaining uninterrupted service provision, even in the presence of hardware malfunctions, network disruptions, or software inconsistencies. Fault tolerance plays a pivotal role in ensuring the dependability and availability of a system, hence becoming an essential component in this context.

The use of distributed systems in cloud computing represents a notable paradigm change in our approach to enhancing fault tolerance and dependability in cloud-based ecosystems. Distributed systems are well-suited for tackling the difficulties related to fault tolerance due to their intrinsic ability to divide workloads and data over different nodes or resources. This methodology is predicated around the utilization of redundancy, replication, and the capacity to effortlessly restore operations following interruptions. The enhancement of cloud services is not only achieved, but it also results in the optimization of resource consumption and scalability. Through the utilization of distributed systems, cloud providers and organizations may ensure the continuous accessibility and functionality of their critical applications and data, even in the presence of faults or failures in individual components or nodes. This article aims to explore in-depth the novel methodologies and frameworks that utilize distributed systems for the development of fault-tolerant cloud computing architectures. It will demonstrate their significant impact on the whole landscape of cloud computing, highlighting its transformational nature.

The field of fault tolerance strategies in cloud computing encompasses a wide range of approaches, including but not limited to redundancy, replication, load balancing, and gentle degradation. The use of these measures is crucial in the effort to reduce interruptions and minimize the consequences of failures on cloud-based applications and services. A fault-tolerant system that exhibits robustness not only serves to enhance the user experience, but also provides protection against potential data loss and financial losses that may arise due to periods of system unavailability.

In order to attain fault tolerance in the realm of cloud computing, a multitude of frameworks and tools have surfaced, each meticulously crafted to tackle distinct issues and meet unique criteria. The objective of this study is to present a thorough examination of these frameworks, explaining their structures, functionalities, and practical implementations. This study investigates the role of distributed systems, including container orchestration platforms like Kubernetes, fault-tolerant storage solutions like Ceph, and serverless computing frameworks like AWS Lambda, in enhancing the fault tolerance of cloud computing environments.

2.Background of cloud computing

In the realm of cloud computing and distributing systems, various studies have delved into the complexities of fault tolerance and load balancing, highlighting their crucial roles in maintaining efficient and reliable cloud services. Nazari Cheraghlou et al. (2016) provided an exhaustive analysis of fault tolerance architectures, underscoring the inherent advantages and challenges of cloud computing, particularly in fault tolerance. The study conducted a thorough examination of various architectures and approaches, with a particular focus on proactive and reactive fault tolerance solutions (Nazari Cheraghlou et al., 2016).

In a study conducted by Kumar and Sharma in 2016 examined fault tolerance methodologies and models within the context of cloud computing. They emphasized the significance of proactive and reactive fault tolerance policies in addressing hardware and software failures. Farther more The study also encompassed an extensive analysis of cloud computing models such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), as well as the importance of fault tolerance within these models (Kumar & Sharma, 2016). The study conducted by Shahid and his collogues in 2020 placed emphasis on the crucial element of load balancing within the context of cloud computing, highlighting its interconnectedness with fault tolerance. The study put out a novel load balancing method that incorporates fault tolerance criteria, therefore making progress in mitigating the current deficiencies in the efficiency of cloud computing (Shahid et al., 2020). The proactive fault tolerance method, proposed by Attallah in 2021, was designed with the objective of improving the dependability and availability of cloud computing infrastructures. The author emphasized a specific focus on the management of virtual machine CPU problems by employing smart load balancing and migration approaches (Attallah et al., 2021).

Haji and his collogues in 2020 presented a thorough examination of dynamic resource allocation techniques within the context of cloud computing. They placed particular emphasis on the significance of effective resource allocation in improving both the Quality of Service (QoS) and cost-efficiency aspects of cloud computing. The study conducted a comprehensive examination of several resource allocation strategies, such as green-cloud and mobile cloud computing, with the objective of enhancing resource use (Haji et al., 2020). The authors Rezaeipanah et al. (2022) introduced a novel approach to enhance fault tolerance in cloud computing by utilizing fuzzy logic. The primary objective of their technique is to enhance the early identification and handling of problems in cloud computing systems. The study conducted by Rezaeipanah et al. (2022) emphasized the potential of fuzzy logic in enhancing the stability and reliability of cloud computing environments (Rezaeipanah et al., 2022). Tang (2022) proposed a fault-tolerant workflow scheduling system for multi-cloud settings, with the aim of enhancing the efficiency and dependability of scientific applications by optimizing execution costs. The framework under consideration encompasses a range of billing mechanisms and incorporates a unique algorithm to optimize cost-effectiveness and dependability in the execution of applications (Tang, 2022). In their recent publication, Zhuang et al. (2021) introduced Hoplite, an innovative collective communication layer specifically developed for distributed systems that operate based on task-oriented principles. According to Zhuang et al. (2021), the aforementioned layer effectively tackled the obstacles associated with collective communication in dynamic contexts and notably enhanced the efficiency and robustness of these systems (Zhuang et al., 2021). In the recent study, Saxena and Singh (2022) introduced OFP-TM, a novel model that combines online virtual machine (VM) failure prediction with failure tolerance mechanisms in order to improve the dependability of cloud computing systems. The aforementioned approach demonstrated noteworthy enhancements in availability and a decrease in instances of VM migration, hence highlighting its potential in optimizing cloud computing (Saxena & Singh, 2022).

Cotroneo et al. (2023) proposed a unique methodology for detecting failures in cloud computing systems. Their strategy involves the utilization of non-intrusive event tracing to enhance the accuracy and reaction time of failure detection in platforms such as OpenStack (Cotroneo et al., 2023). In another study, Ragmani et al. (2020) employed artificial neural networks as a means to forecast probable faults in cloud infrastructure, hence augmenting the overall dependability and effectiveness of the system. The research conducted by Ragmani et al. (2020) played a crucial role in showcasing the practicality of artificial neural networks (ANNs) in fault-tolerant cloud computing (Ragmani et al., 2020). The authors Sathiyamoorthi et al. (2021) introduced an adaptive approach to fault-tolerant scheduling in cloud computing, with a specific emphasis on enhancing Quality of Service. This approach takes into account the failure rate and current workload of resources in order to optimize resource allocation (Sathiyamoorthi et al., 2021). The Threshold-Based Adaptive Fault Tolerance technique was presented by Rawat et al. (2023) as a means to enhance the dependability of cloud services by integrating proactive and reactive tactics. This technique dynamically chose suitable fault tolerance actions based on circumstances and thresholds (Rawat et al., 2023). In their comprehensive study, Jhawar and Piuri (2017) presented a thorough examination of fault tolerance solutions in the realm of cloud computing. The authors delved into many categories of failures and explored a range of methodologies employed to provide fault tolerance. The study conducted by Jhawar and Piuri (2017) was notable due to its thorough examination of strategies aimed at improving the dependability and accessibility of cloud services (Jhawar & Piuri, 2017). The study done by Rehman et al. (2022) involved a comprehensive investigation of fault-tolerance in cloud computing. The researchers explored both proactive and reactive strategies, emphasizing the necessity for continued exploration in fault-tolerance designs and frameworks (Rehman et al., 2022). The study conducted by Mohammadian et al. (2022) investigated fault-tolerant load balancing techniques in the context of cloud computing. The researchers analyzed a range of algorithms, considering factors such as scalability, reliability, and resource usage, among other criteria (Mohammadian et al., 2022). In their recent study, Alaei et al. (2021) proposed a novel approach for scientific workflow scheduling in cloud environments. The authors developed an adaptive defect detection mechanism by leveraging an Improved Differential Evolution algorithm. According to Alaei et al. (2021), the use of this particular method has demonstrated notable enhancements in both scheduling performance and fault tolerance inside cloud computing settings (Alaei et al., 2021).

In the study by Dias in (2021) proposed a cost-effective approach to condition monitoring in rotating machines. Their methodology involved the utilization of cloud-based infrastructure and data mining techniques to identify faults and classify the operational status of these machines (Dias et al., 2021). In their study, Rong et al. (2020) introduced a conceptual model that tackles the issues of data security and result integrity in cloud-based data mining inside hybrid cloud environments. Their approach focuses on outsourced, fault-tolerant, and privacy-preserving frequent itemset mining (Rong et al., 2020). In the study by Shahid et al. (2021) undertook a thorough examination of fault tolerance within the context of cloud computing. They classified various strategies into three categories: reactive, proactive, and resilient ways. The authors also explored the respective functions of these techniques in guaranteeing the dependability and effectiveness of cloud computing systems (Shahid et al., 2021). The study conducted by Li et al. (2021) centered on the examination of fault-tolerant scheduling techniques specifically designed for scientific processes inside cloud settings. The researchers introduced the Real-time and Dynamic Fault-tolerant Scheduling algorithm as a means to effectively handle different forms of resource failures (Li et al., 2021). The study conducted by Hamouda et al. (2021) focused on the design and evaluation of a flexible and resilient framework for hybrid cloud systems. The researchers placed particular emphasis on the implementation of fault detection mechanisms and self-recovery techniques (Hamouda et al., 2021). In their study, Araujo Neto et al. (2019) proposed a fault-tolerant architecture called MULTS, which aims to leverage transient servers in cloud computing. The design incorporates a scenario-based checkpointing method to assure the continuity of operating operations while minimizing costs (Araujo Neto et al., 2019). In their recent study, Zhang et al. (2022) introduced an advanced fault-tolerant framework for fog computing systems. The proposed model utilizes a Markov chain-based methodology to enhance system performance by dynamically assessing the dependability of fog nodes (Zhang et al., 2022). Chang et al. (2014) introduced a fault-tolerant load balancing strategy for web services, with a focus on implementing replication across many servers to minimize the potential for data loss (Chang et al., 2014). In a similar vein, Chang et al. (2014) and Fang et al. (2019) integrated fault tolerance and workload balancing inside the Distributed Stream Processing Engine to effectively handle dynamic situations including data asymmetry and node failures. Fang et al. (2019) have made a significant addition by proposing a proactive fault tolerance strategy that specifically addresses the implementation and selection of cloud servers to mitigate network congestion. Their research demonstrates that this approach leads to decreased overhead and energy usage (Fang et al., 2019). The fault-tolerant workflow scheduling in large-scale scientific applications proposed by Nirmala et al. (2020) incorporates the utilization of unsupervised learning techniques to enhance replication heuristics, as discussed by (Setlur et al., 2020). The GBFTLB technique was proposed by Chatterjee et al. (2020). This algorithm is designed to address load balancing in heterogeneous processors, with a focus on minimizing communication costs and accommodating various network topologies.

Chatterjee et al. (2020) have developed approaches such as Imitator for graph computing (Chatterjee et al., 2020). These techniques employ vertex replication to ensure fault tolerance, while also including fast recovery mechanisms and minimizing execution cost. Various proactive strategies have been investigated in the context of virtualized cloud federation setups, with a specific focus on managing CPU temperature to mitigate node failures. These techniques aim to optimize profitability and minimize migration costs. Researchers have put forth collaborative checkpoint-rollback recovery techniques as a means to mitigate communication costs in dynamic distributed platforms, therefore decreasing the need for coordinating message exchanges. The ReadyFS method, which has been specifically developed for the purpose of scheduling scientific processes in cloud environments, enhances the efficiency of resource allocation while simultaneously guaranteeing fault tolerance and adherence to deadlines. The utilization of adaptive approaches that employ fuzzy-based methods for failure detection in cloud computing, together with the implementation of aggressive fault tolerance approaches employing intelligent decision agents, serves as a notable demonstration of the progress made in this domain. These improvements contribute to the improvement of performance and the reduction of complexity when compared to conventional methodologies. The aforementioned research collectively demonstrate a substantial advancement in improving fault tolerance and load balancing in cloud computing. They showcase a variety of techniques and strategies aimed at tackling these crucial difficulties.

2.1. Cloud computing infrastructure:

The cloud computing infrastructure encompasses a range of essential components, such as computers, storage devices, network facilities, and other related elements, that are required to provide customers with cloud computing resources and services(Salih Ageed et al., 2021). The aforementioned hardware components are predominantly situated within business data centers (Al-Jaroodi et al., 2012). The mentioned components include multicore servers, solid-state drives, and hard disc drives, which provide reliable storage, as well as network equipment like firewalls, switches, and routers, all operating at a significant scale. In addition to the aforementioned physical components, the software components that facilitate the cloud service model, including virtualization software, are also referred to as the infrastructure of cloud computing. The virtualization software facilitates the creation of a conceptual representation of cloud resources and makes these resources accessible to users through the use of APIs (Application Programme Interfaces) or other interfaces, such as command line or graphical interfaces. The cloud service providers (CSPs) often supply virtualized resources to consumers over the Internet or other networks (Bala & Chana, 2012).

Cloud computing resources are commonly provided to users in the form of services, typically following a shared and multitenant-based model. Major cloud service providers such as Amazon Web Services (AWS) and Google Cloud Platform employ a multitenant-based model. This methodology is employed to facilitate the sharing of resources in a manner that is both cost-effective and secure across various applications and tenants, such as corporations and organisations, inside the cloud computing environment. Virtualization software may be employed to guarantee the establishment of isolation between many tenants. A conventional cloud architecture consists of several components, including clients, servers, applications, and other relevant elements (Mohammed et al., 2017).

Another essential component of cloud computing is the distributed file system (DFS), such as the Google File System (GFS) and/or Hadoop Distributed File System (HDFS), which primarily serves the purpose of storing data on discs in the form of objects or blocks. These file systems separate the administration of storage from the physical storage itself, hence guaranteeing the scalability of storage. Therefore, the cloud computing infrastructure encompasses, in a general sense (Singh et al., 2016), the following components:

- Servers: are physical devices that serve as host machines for one or more virtual computers.

- Virtualization: is a technological process that involves the abstraction of physical components, such as servers, storage, and networking, and presents them as logical resources.
- Storage: encompasses several technologies such as Storage Area Networks (SAN), network attached storage (NAS), and disc drives, among others. Additionally, it includes functionalities such as archiving and backup.
- Network: is established in order to facilitate the connecting of physical servers and storage devices.
- Management: encompasses a range of software applications designed to configure, administer, and monitor cloud infrastructure components such as servers, networks, and storage devices.
- Security: refers to the several elements that ensure the integrity, availability, and confidentiality of data, as well as the overall protection of information.
- Backup: The provision of backup and recovery services.

2.2. Cloud deployment models

The cloud deployment model is determined by the purpose and context in which a cloud service is intended to be utilised. The choice of the deployment model has a direct impact on several financial aspects, such as the cost spent, power consumption of resources, and other capital costs (Haji et al., 2020)The deployment methods most frequently employed in cloud settings are :

The public cloud: refers to a computing infrastructure that allows unrestricted access to systems and services provided by an enterprise provider for the general public. The use of multi-tenancy in cloud computing allows for cost-effective solutions that offer flexibility, scalability, and location independence (Jain et al., 2014; Nazari Cheraghlou et al., 2016; Singh et al., 2016). Resources are allocated in a flexible manner, based on immediate need, from an external service provider who use a multi-tenant framework to deliver these resources.

- The private cloud: refers to the utilization of cloud resources and services exclusively within a certain organization. In other words, access and usage of the cloud resources are limited to internal use within the organisation. This paradigm guarantees robust application and data security and privacy, as supported by previous studies (Jain et al., 2014; Nazari Cheraghlou et al., 2016; Singh et al., 2016).
- The Community Cloud: concept is employed by several firms and organisations concurrently to support a certain community or society, addressing communal needs such as security requirements, mission objectives, and compliance considerations, among others. This approach has the potential to be operated, owned, and controlled by either one or numerous organisations inside the community, or alternatively, by a third party.
- The hybrid cloud: refers to a strategic combination of public and private cloud environments. In this cloud deployment, the execution of essential events, particularly those that necessitate secure operations, is facilitated through the utilisation of private cloud services. On the other hand, non-critical events are carried out by using the capabilities of the public cloud.

2.3. Cloud service model:

Cloud computing has seen significant advancements in recent years; yet, it is still categorized under three primary service types (Haji et al., 2020).Figure 1 illustrates the fundamental service models:

- Software-as-a-Service (SaaS): refers to the provision of software applications as services by cloud service providers to customers or end-users. The provision of a cloud application as a service streamlines maintenance procedures by eliminating the need for users to install and operate the program on their own machines. Some examples of communication methods in the digital age include online conferencing, electronic mail (email), and social networking platforms. Amazon AWS, Google Compute Engine, Microsoft Azure, IBM SmartCloud Enterprise, CloudStack, OpenStack, Open-Nebula, CloudForge, Citrix, Qstack, and several other platforms are recognised as suppliers of Software-as-a-Service (SaaS).
- Platform-as-a-Service (PaaS): is a cloud computing model that facilitates the development, deployment, and management of applications. It offers users the ability to construct, operate, evaluate, and oversee cloud-based applications. Custom application development can be carried out within a software stack environment that is rented from a cloud service provider (CSP). Several well-known suppliers offer Platform as a Service (PaaS) solutions, including Acquia Cloud, Amazon AWS, App Agile, Apprenda, AppScale, Bluemix, Cloud 66, Cloudways, and other others.

- Infrastructure-as-a-Service (IaaS): refers to a paradigm that grants users access to tangible computing resources such as computers, storage systems, networks, servers, and virtual machines hosted on cloud platforms. The Infrastructure as a Service (IaaS) provider offers storage resources that may be used as needed, as well as the ability to create and manage virtual machines in a flexible manner. Salesforce, Microsoft, Amazon Web Services (AWS), Slack, Zendesk, GitHub, Oracle, Cisco, and several more companies offer Software-as-a-Service (SaaS) solutions .
- The concept of Everything-as-a-Service (XaaS) encompasses a wide range of services that may be delivered in various forms. Singh et al. (2016) assert that the cloud system possesses the capability to effectively handle substantial quantities of resources tailored to meet specific requirements. These resources encompass Security-as-a-Service, Identity-as-a-Service, Communication-as-a-Service, DaaS, and Strategy-as-a-Service.

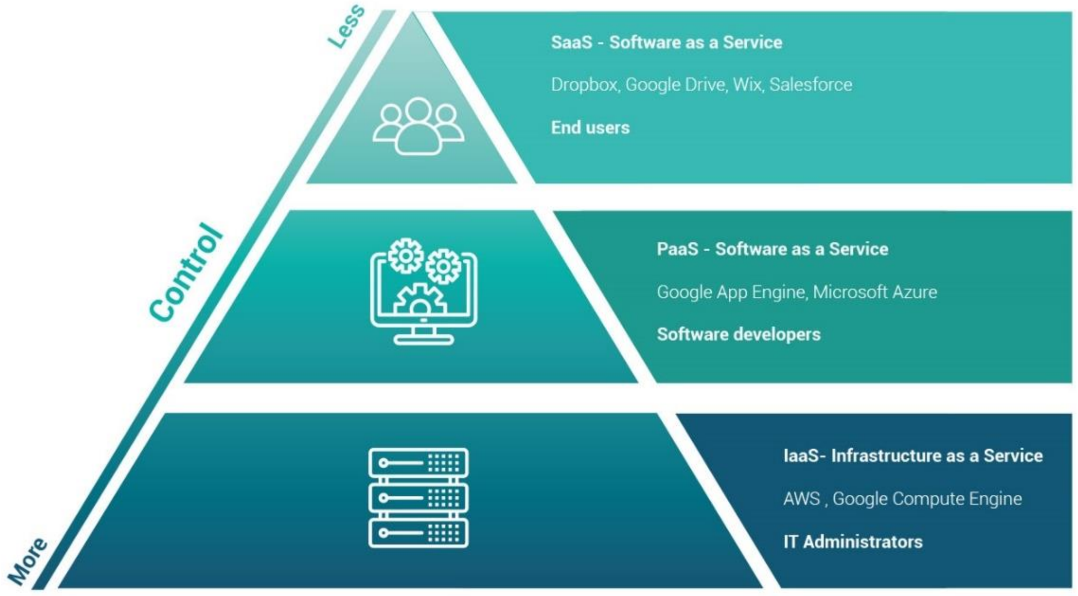
Less
Control
More
SaaS - Software as a Service
Dropbox, Google Drive, Wix, Salesforce
End users
PaaS - Software as a Service
Google App Engine, Microsoft Azure
Software developers
IaaS- Infrastructure as a Service
AWS , Google Compute Engine
IT Administrators

Figure 1:Cloud service model

3. Leveraging Distributed system in Cloud Computing:

Understanding Distributed Applications in Cloud Computing provides a comprehensive exploration of distributed applications within the context of cloud computing, by emphasizing the growing popularity of distributed applications, which are designed to operate across multiple computers or servers. These applications leverage parallel processing to enhance system performance, making them a pivotal aspect of modern cloud computing environments (Zankoya Zaxo et al., 2018). The subject delves into various crucial elements such as fault tolerance, scalability, data consistency, and synchronization(Al-Jaroodi et al., 2012; Poola et al., 2017).

1. Understanding Distributed Applications:
   - Distributed applications are increasingly popular in cloud computing, designed to run on multiple computers or servers.
   - They divide tasks among multiple machines for parallel processing, improving system performance.
   - Key considerations include fault tolerance, reliability, scalability, data consistency, and synchronization.
   - Case studies (e.g., Netflix and Airbnb) illustrate successful implementations of distributed applications.
2. Benefits of Leveraging Cloud Computing:
   - Cost Efficiency: Reduces hardware and infrastructure costs with a pay-as-you-go model.
   - Scalability and Flexibility: Allows quick scaling of resources to meet changing demands.
   - Reliability and Availability: Ensures high availability through robust infrastructure and redundancy measures.
   - Enhanced Security: Advanced security measures protect infrastructure and customer data.
   - Improved Collaboration and Accessibility: Facilitates seamless collaboration and remote access.

- Case Study: Netflix demonstrates successful cloud migration for scalable, cost-efficient streaming services.

3. Key Components of Cloud Computing for Distributed Applications:
- Scalability: Essential for handling varying workloads and traffic.
- Elasticity: Automatically provisions and deprovisions resources as needed.
- Fault Tolerance: Ensures high availability and resilience to failures through mechanisms like data replication and load balancing.
- Security: Provides robust protocols, encryption, and access controls to safeguard sensitive information.
- Cost Efficiency: Offers a pay-as-you-go model to optimize resources and reduce costs.
- Interoperability: Enables seamless data exchange and integration with various systems and services.

4. Fault tolerance approaches in distributed systems:

The existence of component failures or numerous faults necessitates fault tolerance in a system, which is essential for enabling the system to provide the required services. System failures are a consequence of mistakes that arise from underlying flaws (see Figure 4). The following descriptions are provided (Poola et al., 2017):
- Faults: refer to the inability of a system to carry out its essential tasks due to the presence of aberrant states or bugs inside one or more components of the system The system may experience several sorts of error.
- Error: The existence of faults can cause a system component to enter an error state or wrong condition. When a system component performs incorrectly, it can cause the system to fail partially or completely.
- Failure: The concept of failure is a significant topic of discussion and analysis in various academic disciplines. It The term "system misbehavior" pertains to the occurrence of anomalous actions within a system, which can be detected by a user, whether human or another computer system. The recognition of a failure occurs just when the output or outcome of a system is deemed erroneous. Failures can be categorized according to the classification given in Figure 2.

Fault
Error
Failure
Figure 2: failure steps

Fault tolerance techniques are required because they help in the detection and treatment of system faults caused by failure either hardware or software. Fault tolerance is extremely important in cloud platforms because it ensures application performance, dependability, and availability. To achieve resilience in cloud computing, it is necessary to properly access and handle failure(Kathpal & Garg, 2019) Some fault tolerance techniques discovered from literature may be classified as follows Figure 3 (Mukwevho & Celik, 2021):
- Reactive Fault Tolerance: This method focuses on mitigating the impact of failures in cloud systems after they have occurred. It enhances the robustness and reliability of the system and has been applied in both cloud and other distributed systems.

Proactive Fault Tolerance: This strategy proactively identifies potential faults and replaces components at risk of failure with operational ones, thereby preventing faults and errors rather than recovering from them,
- Fault Tolerance Parameters in Cloud Computing: Different parameters are used to assess the efficiency and effectiveness of fault tolerance approaches in cloud computing systems (Rahman & Rouf, 2022).These parameters, crucial for evaluating cloud system performance includes Adaptive, Response Time, Performance, Throughput, Reliability, Availability, Usability, Overhead.

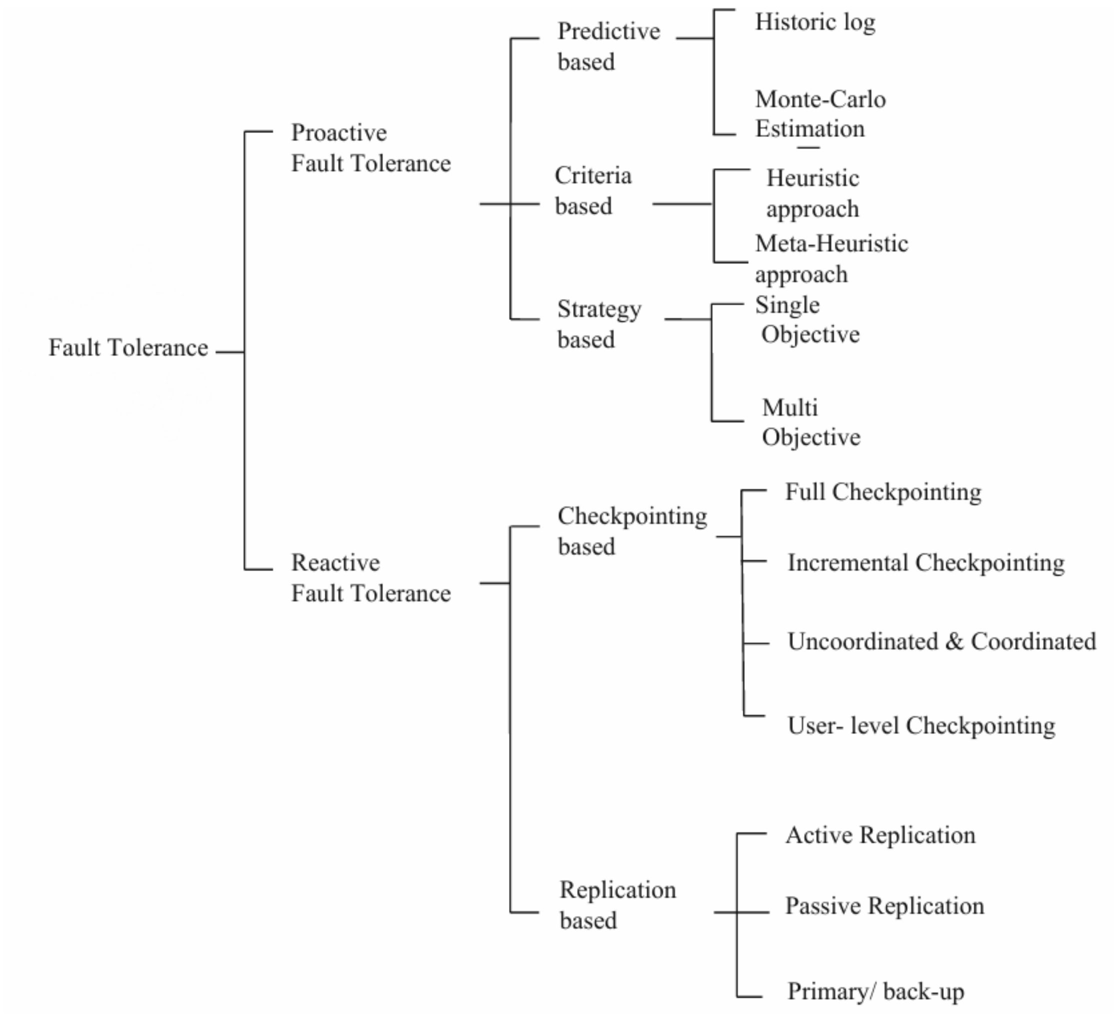

Figure 3:Fault Tolerance Classification(Poola et al., 2017)

Table 1, provides an overview of common failures in computing systems, categorizing them into hardware, software, and network errors. Hardware errors, caused by factors like external radiations and aging, are measured using metrics like FIT and MTTF and are typically mitigated through spatial and data redundancy techniques such as TMR and RAID configurations. Software errors, arising from issues like bugs and poor design, are quantified by metrics including the number of bugs per thousand lines of code and are addressed using approaches like N-version programming and checkpoint-based recovery.

Network errors, resulting from congestion or attacks, are measured by packet loss rates and similar metrics, with solutions encompassing resource reservation and various redundancy methods, including data redundancy like CRC and spatial redundancy like replicated nodes. Each category of error highlights specific causes, measurement metrics, and traditional mitigation strategies, providing a comprehensive view of fault tolerance in computing environments.

Table 1:common failures in computing systems

	Failures	Causes	Metrics	Traditional Approaches
Hardware Errors	Soft Errors, Hard Failures, System Crash	External Radiations, Thermal Effects, Power Loss, Poor Design, Aging	FIT, MTTF, MTBF	Spatial Redundancy (TMR, Duplex, RAID-1 etc.) and Data Redundancy (EDC, ECC, RAID-5, etc.)
Software Errors	Wrong outputs, Infinite loops, Crash	Incomplete Specification, Poor software design, Bugs, Unhandled Exception	Number of Bugs/Klines, QoS, MTTF, MTBF	Spatial Redundancy (N-version Programming, etc.), Temporal Redundancy (Checkpoints and Backward Recovery, etc.)
Network Errors	Data Losses, Deadline Misses, Node (Link) Failure, System Down	Network Congestion, Noise/Interference, Malicious Attacks	Packet Loss Rate, Deadline Miss Rate, SNR, MTTF, MTBF, MTTR	Resource Reservation, Data Redundancy (CRC, etc.), Temporal Redundancy (Retransmission, etc.), Spatial Redundancy (Replicated Nodes, MIMO, etc.)

5. Frameworks of fault tolerance:

Computer fault tolerance frameworks ensure continued and reliable operation despite mistakes or failures. These frameworks include tactics and technologies to identify, handle, and recover from hardware, software, and network failures(Amoon, 2016). These frameworks minimize downtime and protect system integrity by using redundancy and failover strategies. They also use predictive tactics to prevent failures and real-time monitoring to fix problems. These frameworks provide a solid architecture to maintain operational continuity, data integrity, and service availability in essential applications including cloud computing and real-time data processing systems (Karande & Pais, n.d.). Table 2, shows Characteristics and Limits of some Fault Tolerance Frameworks.

Table 2: Fault Tolerance Frameworks - Characteristics and Constraints

Framework	Characteristics	Constraints
AITRC	Ensures real-time application accessibility and test consistency	Resource availability may decrease with increased load
BlobCR	Offers dynamic continuous snapshots with low overhead	Inapplicable for loosely coupled applications due to potential separation
BITCloud	Highly resilient, handles all Byzantine faults	Inefficient capital utilization
AASIF	Effective resource utilization	Fault tolerance production may be inefficient despite its adaptive approach
DAIT	Addresses all incident defects, optimized overhead with robust design	Device-dependence limits adaptability, potentially affecting defect sensitivity
FfCloud	Ideal for detecting program flaws due to high sensitivity	Complex design requirements and inefficient asset use
CAMAS	Adaptable, fast fault resilience with low operational costs	Limited to resolving asset dispute defects

ITM	Independent from provider, applicable to all accident defects	Relatively high fault tolerance overhead
FTWS	High resource utilization, suited for modeling various failures	Not suitable for all cloud environments due to lack of location instancing
FESTAL	Energy-efficient with strong resource usage	Performance issues if both primary and reserve systems fail simultaneously

Fault tolerance techniques are required because they help in the detection and treatment of system faults caused by failure The presented table offers a comprehensive summary of diverse reactive fault tolerance systems, emphasizing their distinct attributes and possible limitations within the realm of cloud computing.

Table 3, categorizing different fault tolerance strategies provides a comprehensive overview of how systems can maintain functionality despite errors or malfunctions. Each strategy in the table is described in terms of its approach, highlighting key characteristics and methods of implementation. The table also outlines the advantages ("Pros") and disadvantages ("Cons") of each strategy, offering insights into their effectiveness, resource requirements, and potential impact on system performance. For example, redundancy strategies might be highly effective in preventing data loss (a pro) but could be resource-intensive (a con). Furthermore, the table includes specific "Use Cases" for each strategy, illustrating the scenarios or system environments where they are most applicable. This might include high-availability systems, real-time processing environments, or data-sensitive applications. Such a table serves as a valuable resource for system designers and engineers, helping them choose the most appropriate fault tolerance methods for their specific needs and constraints.

Table 3: categories of fault tolerance strategies

Strategies	Description	Pros	Cons	Use Cases
Redundancy-Based	Utilizes redundant resources and components to ensure availability.	- High availability - Immediate fault recovery	- Increased cost - Resource wastage	Critical applications, financial services
Replication-Based	Duplicates data and services across multiple locations or instances.	- Data consistency - Low-latency access	- Complexity in synchronization - Increased storage costs	Content delivery, database systems
Load Balancing	Distributes incoming traffic across multiple servers or instances.	- Improved resource utilization - Scalability	- Single point of failure (load balancer)	Web applications, e-commerce sites
Checkpoint-Restart	Periodically saves application state and restarts from the last checkpoint in case of failure.	- Fast recovery - Minimal data loss	- Increased I/O overhead - Limited support for stateful applications	Scientific simulations, batch processing
Auto-Scaling	Dynamically adjusts resource allocation based on traffic or demand.	- Cost-effective - Scalability	- Requires proper configuration - Potential resource underutilization	Web services, IoT applications
Microservices	Decomposes applications into small, independent services that can be individually scaled and managed.	- Flexibility - Isolation of faults	- Complex to implement and manage - Network latency	Containerized applications, cloud-native services

Chaos Engineering	Proactively injects faults and failures to test system resilience.	- Identifies weaknesses - Improves overall system reliability	- Potential service disruption - Resource-intensive	Highly resilient applications, large-scale systems

Table 4 presents a comparative analysis of 20 prior investigations. This paper provides a comprehensive analysis of many research endeavors pertaining to fault tolerance in distributed and cloud systems. Every study is distinguished by its reference (Ref.) and is examined across many critical dimensions. The column labeled "Main Focus" serves to emphasize the primary subject or issue that each study aims to address, whereas the "Key Techniques" column provides a comprehensive account of the specific methodologies or approaches utilized in the research. The section under "Tools/Models Used" offers an elucidation of the practical or theoretical frameworks employed in the research. The efficacy and influence of these investigations are assessed through the utilization of "Evaluation Metrics," while the "Outcome/Results" section succinctly presents the results or conclusions derived from the research.

The section under "Challenges Addressed" provides an elucidation of the particular obstacles or areas of limited understanding that the research endeavor to surmount. The column labeled "Innovations/Contributions" delineates the distinctive contributions made by each study to the subject. On the other hand, the "Future Directions" column proposes prospective avenues for further research or more advancements. This comprehensive table functions as a significant tool for comprehending the extent, approach, and influence of various studies, offering a full perspective of the research environment in the topic under discussion.

Table 4: Previous Study comparison

#	Ref.	Main Focus	Key Techniques	Tools / Models Used	Evaluation Metrics	Outcome / Results	Challenges Addressed	Innovations / Contributions	Future Directions
1	(Cotroneo et al., 2023)	Fault tolerance in fog computing	Dynamic VM migration, load balancing	Fuzzy logic, CloudSim	Execution time, VM utilization, system stability	Improved fault tolerance and load balancing	Network and hardware failures, resource allocation	Load-balanced and fault-tolerant fog computing architecture	Integration with IoT, optimization strategies
2	(Ragmani et al., 2020)	Predictive analytics for fault prevention in cloud	Artificial neural network for fault prediction	Weka toolkit, Backblaze datasets	Accuracy, correctly classified instances, error rates	Accurate failure prediction, reduced failures and costs	Data access limitations, large data size	ANN module for fault prediction, VM allocation failures forecast	Real-time assessment, CloudSim integration
3	(Sathiyamoorthi et al., 2021)	Adaptive fault-tolerant scheduling in cloud	Proactive fault-tolerant resource allocation	CloudSim, AFTRA system	Resource utilization, execution time, makespan, success rate	Enhanced QoS, efficient than benchmark techniques	Combining fault tolerance with load balancing	Adaptive Fault Tolerant Resource Allocation (AFTRA) approach	Optimization algorithms, cost, security, energy consideration
4	(Rawat et al., 2023)	Threshold-based adaptive fault tolerance in cloud	Proactive migration, replication, reactive retry	CloudSim, TBAFT model	Total execution time, migration time, failure rate, throughput	Reduced failure rate, improved cloud application resiliency	Using both proactive and reactive methods effectively	Threshold-Based Adaptive Fault Tolerance (TBAFT) approach	Reducing fault tolerance complexity, broader application scenarios
5	(Rehman et al., 2022)	Fault-tolerance methods, architectures, and frameworks in cloud computing	Proactive and reactive approaches to fault-tolerance, including hardware and software fault-tolerance	AFTRC model, SVM, DAFT	Mean Time to Failure (MTTF), Mean Time to Repair (MTTR), Availability	Fault-tolerance in MapReduce, HAProxy for load balancing and fault-tolerance	Complexity in designing fault-tolerance for varied cloud architectures	Detailed survey on fault-tolerance methods and architectures in cloud computing	Developing more efficient fault-tolerance strategies and architectures
6	(Mohammadian	Systematic literature review of	Comprehensive review of centralized	CloudSim	Accuracy, fault tolerance	Comprehensive overview of fault-tolerant	Lack of organized studies on	Systematic review of centralized and distributed fault-	Exploration of new fault-tolerant load

	et al., 2022)	fault-tolerant load balancing in cloud computing	and distributed fault-tolerant load balancing methods			load balancing methods	fault-tolerant load balancing methods	tolerant load balancing algorithms	balancing methods and techniques
7	(Alaei et al., 2021)	Adaptive fault detection strategy for scientific workflow scheduling in cloud computing	Improved Differential Evolution (IDE) algorithm, proactive and reactive fault tolerance strategies	Adaptive Network-Based Fuzzy Inference System (ANFIS), CloudSim	Makespan, energy consumption, total cost, fault ratio	Improved scheduling performance, higher degree of fault tolerance, minimized energy consumption and cost	High failure rates in cloud computing, inefficiencies in traditional fault-tolerant methods	Adaptive fault detector strategy, integration of proactive and reactive fault tolerance methods	Improving fault tolerance in cloud computing with advanced algorithms and techniques
8	(Dias et al., 2021)	Fault detection and identificatio n in rotating machines using cloud-based condition monitoring	SVM and OCSVM for operation status classification	Feature selection strategies like CFS, ACFS, and AutoEncoder	Accuracy, False Positive Rate (FPR), False Negative Rate (FNR), Execution time	High accuracy (87.5% to 100%) and robustness, reduced execution time	Low-cost solution for small and medium enterprises without requiring dedicated sensors	A new strategy for feature selection, use of SVM and OCSVM for fault detection	Improvements in machine learning algorithms for better fault detection and identification
9	(Rong et al., 2020)	Secure and fault-tolerant frequent itemset mining in a hybrid cloud environmen t	Shamir's secret sharing for privacy-preserving and integrity verification	Shamir's secret sharing scheme for data outsourcing and integrity verification	Security, fault tolerance, efficiency, error detection and correction capabilities	Efficient protocol, protection against frequency analysis attack, integrity verification	Security concerns in data mining outsourcing, unauthorized data breaches	Privacy-preserving building blocks and secure data outsourcing scheme	Enhancing the protocol for better efficiency and security in diverse cloud environments
10	(Shahid et	Comprehen sive survey of fault	Reactive, Proactive, and Resilient	Various models and frameworks for	Comparative study of various fault tolerance	Detailed categorization and analysis of	Heterogeneity, automation needs,	Comprehensive survey and classification of fault	Further research in AI and machine

	al., 2021)	tolerance methods in cloud computing	methods for fault tolerance	different fault tolerance methods	frameworks and their effectiveness	fault tolerance methods and their applications	recovery objectives, workloads in the cloud	tolerance methods, introduction of resilient methods	learning for adaptive fault tolerance and system learning
11	(Li et al., 2021)	Fault-tolerant scheduling for scientific workflows in cloud environments	Real-time and dynamic fault-tolerant scheduling (ReadyFS) algorithm	ReadyFS algorithm, RDE, CDE, ReSC, and RA strategy	Performance in terms of fault tolerance, resource utilization, execution time	Improved fault tolerance and resource utilization, reduced execution time, successful handling of different failure types	Addressing various types of cloud resource failures, improving fault tolerance while maintaining resource efficiency	Introduction of ReadyFS algorithm, novel fault-tolerant scheduling mechanisms	Further optimization of the ReadyFS algorithm, addressing other cloud computing challenges
12	(Hamouda et al., 2021)	Reconfigurable fault-tolerant and self-recovering systems in hybrid Clouds	Formal modelling and verification using BIP framework, focusing on failure detection and recovery	BIP (Behavior & Interaction & Priority) tool, SBIP (Stochastic BIP) for modelling and verification	Real-time experimentations, formal verification analysis, stochastic formal verification using SBIP model checker	Successful failure detection and recovery, improved reliability and performance of hybrid Cloud systems	Managing failures in heterogeneous and complex hybrid cloud architectures	Novel formal modeling and verification approach for hybrid cloud systems, focusing on reconfigurability and self-recovery	Extending the modeling approach to other types of cloud environments, enhancing the verification process
13	(Rezaeipanah et al., 2022)	Fault tolerance in cloud computing	Fuzzy logic, task resubmission, timing check, migration techniques	Fuzzy fault detection system and fault tolerance increase system	Accuracy, response time, load density, throughput	Mean superiority of 6.49% for accuracy criterion compared to ABFT method	Fault detection in cloud computing VMs	New fuzzy-based method for fault management, improved system reliability without precise input dataset	Probabilistic approach to increase the accuracy
14	(Tang, 2022)	Workflows scheduling on multi-cloud systems	Weibull distribution analysis, fault-tolerant cost-efficient workflow	FCWS algorithm	Task execution reliability, hazard rate	Minimizing application execution cost time while ensuring reliability	Multiple billing mechanisms, virtual resources heterogeneity,	Integrated multi-cloud providers' billing mechanism into scheduling, fault-tolerant cost-efficient workflow scheduling algorithm	Not specified

			scheduling (FCWS)				systems reliability		
15	(Zhuang et al., 2021)	Collective communication efficiency in task-based distributed systems	Distributed scheduling scheme, fine-grained pipeline scheme	Hoplite's object directory service	Not specified	Speeds up asynchronous stochastic gradient descent, reinforcement learning, and serving ensemble of machine learning models	Traditional collective communication libraries' limitations	Efficient and fault-tolerant collective communication layer, data transfer schedules computed on the fly	Not specified
16	(Saxena & Singh, 2022)	VM failure prediction and tolerance in cloud computing environments	Ensemble method-based failure predictor, Failure Tolerance Unit (FTU)	OFP-TM model	Service availability, VM failure reduction, server overload prediction, resource utilization, power consumption, VM migration cost	Improvement in availability, reduction in number of live VM migrations	Cloud outages, SLA violation, excessive power consumption	Online VM failure prediction and tolerance model, proactive detection and handling of resource failures	Not specified
17	(Nazari Cheraghlou et al., 2016)	Fault Tolerance in Cloud Computing	Redundancy techniques, proactive and reactive policies	Map-reduce, FT-Cloud, Haproxy, Byzantine, Gossip, MPI, FTM, LLFT, Magi Cube, Vega Warden, AFTRC, PLR architectures	Comparison of architectures based on policy types and fault detection and recovery methods	Comparison of different fault tolerance architectures	Fault detection and recovery in cloud computing	Detailed comparison and analysis of various architectures	Further research in the field
18	(Kumar & Sharma, 2016)	Fault Tolerance in Cloud Computing	Proactive FT Policy, Reactive FT Policy, Checkpointing /Restart, Job Migration,	Various fault tolerance models like VFT, LLFT, FTM, ASSURE, SHelp, FTMC, AFTRC, FTWS	Reliability, Performance, Response Time	Comparative analysis of different fault tolerance models	Load balancing, fault prediction, and recovery	Comparative analysis and identification of the most efficient fault tolerance techniques	Developing a comprehensive fault tolerance scheme

			Task Resubmission						
19	(Shahid et al., 2020)	Load Balancing and Fault Tolerance in Cloud Computing	Various load balancing techniques, fault tolerance metrics	Sender initiated, receiver initiated, symmetric load balancing approaches, Machine Learning scheduling algorithms	LB parameters like performance, overhead, scalability, fault tolerance	Proposal of a novel algorithm incorporating FT in LB	Load balancing challenges in cloud computing	Introduction of a novel algorithm based on FT metrics in load balancing	Not specified
20	(Attallah et al., 2021)	Proactive Load Balancing and Fault Tolerance in Cloud Computing	Proactive Fault Tolerance Policy, Reactive Fault-Tolerant Policy, Load Balancing	PLBFT model, Ceilometer telemetry service	MTBF, Failure Rate	PLBFT model with higher reliability and reduced failure rate compared to other models	Challenges of fault tolerance in cloud computing	PLBFT model that proactively handles faults and applies load balancing	Extension of the model to include space and memory faults

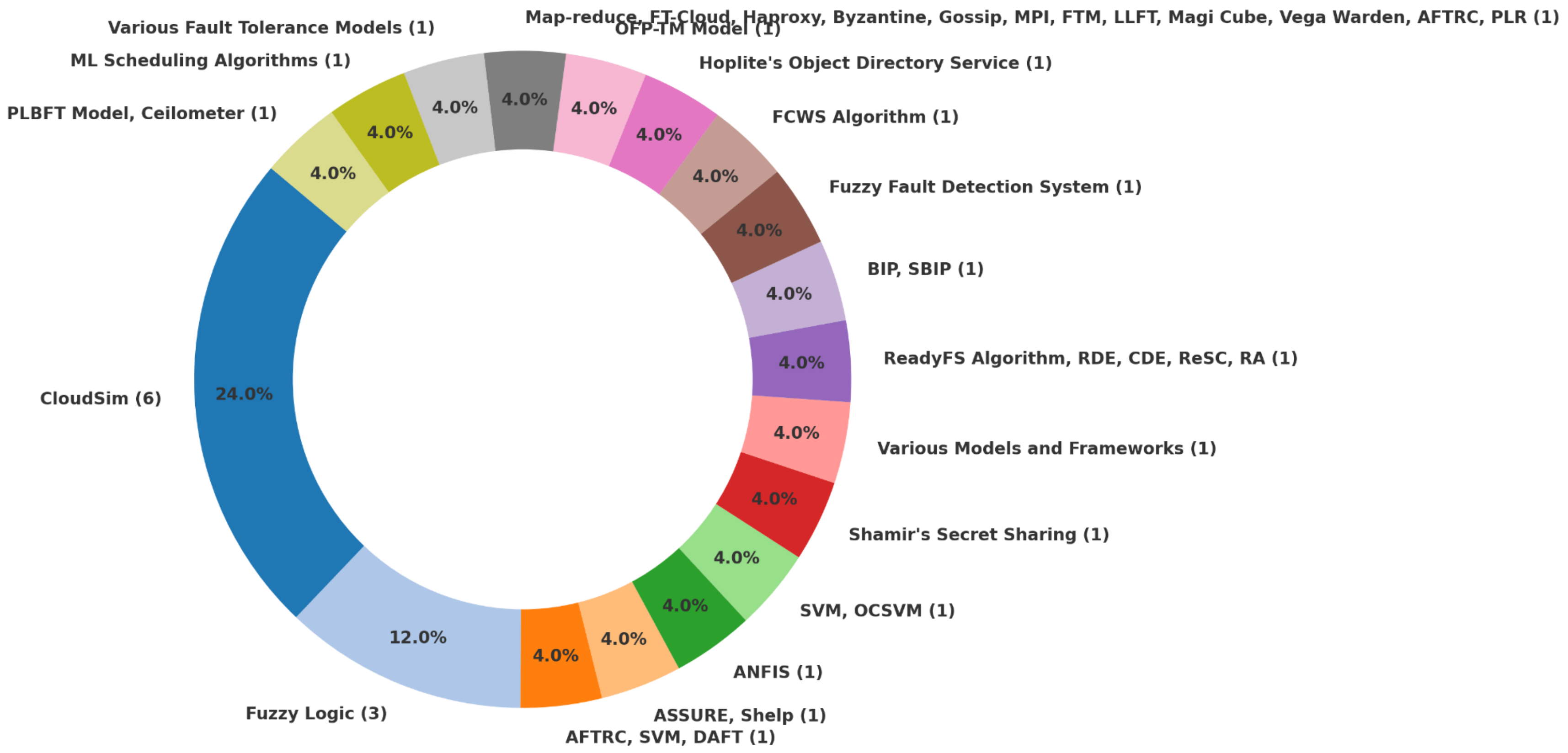

Figure 4: Fault Tolerance Tools used in 20 studies

The presented pie chart provides a concise overview of the allocation of techniques and models employed in fault tolerance research across a variety of scholarly articles. CloudSim is the most prevalent tool in the chart, as it is utilized in 24% of the papers. Following closely behind is Fuzzy Logic, which is applied in 12% of the articles. The work includes references to many tools and models, namely ML Scheduling Algorithms, Various Fault Tolerance Models, the PLBFT Model, and Ceilometer, with each being mentioned for 4% of the total citations. Furthermore, this field of research encompasses a varied spectrum of approaches, as seen by the inclusion of several tools and techniques such as MapReduce, HAProxy, Byzantine systems, Gossip, MPI, FTM, LLFT, Magi Cube, Vega Warden, AFTRC, and PLR, each of which is represented by a single article. The chart illustrates a diverse range of strategies for achieving fault tolerance, with a particular emphasis on the utilization of simulation and fuzzy logic techniques.

7. Conclusion:

This study offers a comprehensive analysis of fault tolerance in the context of cloud computing. This statement underscores the importance of ensuring the consistent and uninterrupted provision of services, even in the event of potential disruptions such as hardware malfunctions, software issues, and network outages. This study highlights the significance of distributed systems in improving the durability and resilience of cloud services through an analysis of various fault tolerance strategies and frameworks. The enhancement of cloud infrastructure is examined through the analysis of strategies such as redundancy, replication, and flawless recovery from disturbances. These strategies are identified as crucial components in strengthening cloud infrastructure.

The last section of this research study underscores the significant and transformational effects that fault tolerance solutions have on the field of cloud computing. This statement emphasizes the need of implementing steps to ensure the continuous functioning of cloud-based apps and services. By doing so, organizations may safeguard themselves against the risks of data loss and financial consequences that may arise from periods of inactivity. The report furthermore presents a comparative analysis of twenty research, elucidating different methodologies employed to improve fault tolerance in cloud systems, ranging from conventional to novel techniques.

The conclusion of the study may emphasize the persistent requirement for further research and development in this field, with a specific emphasis on the incorporation of emerging technologies like artificial intelligence and machine learning. This integration aims to anticipate and proactively mitigate future system failures. Potential future possibilities might involve the advancement of increasingly intricate and adaptable fault tolerance frameworks capable of accommodating the escalating intricacy and heterogeneity observed in cloud computing environments.

References

References

1. 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*. https://doi.org/10.1016/j.asoc.2020.106895
2. Lakhan, A., Mohammed, M. A., Zebari, D. A., Abdulkareem, K. H., Deveci, M., Marhoon, H. A., ... & Martinek, R. (2024). Augmented IoT Cooperative Vehicular Framework Based on Distributed Deep Blockchain Networks. *IEEE Internet of Things Journal*.
3. H. Shukur, S. Zeebaree, R. Zebari, D. Zeebaree, O. Ahmed, and A. Salih, “Cloud Computing Virtualization of Resources Allocation for Distributed Systems,” Journal of Applied Science and Technology Trends, vol. 1, no. 3, pp. 98–105, Jun. 2020, doi: 10.38094/jastt1331.
4. Mohammed Mohammed Sadeeq, Nasiba M. Abdulkareem, Subhi R. M. Zeebaree, Dindar Mikaeel Ahmed, Ahmed Saifullah Sami, and Rizgar R. Zebari, “IoT and Cloud Computing Issues, Challenges and Opportunities: A Review,” 2021, doi: 10.48161/issn.2709-8206.
5. Abdullah, P. Y., Zeebaree, S. R., Jacksi, K., & Zeabri, R. R. (2020). An hrm system for small and medium enterprises (sme) s based on cloud computing technology. International Journal of Research-GRANTHAALAYAH, 8(8), 56-64.
6. Abdullah, P. Y., Zeebaree, S. R., Shukur, H. M., & Jacksi, K. (2020). HRM system using cloud computing for Small and Medium Enterprises (SMEs). Technology Reports of Kansai University, 62(04), 04.
7. Zeebaree, S. R., Zebari, R. R., Jacksi, K., & Hasan, D. A. (2019). Security approaches for integrated enterprise systems performance: A Review. Int. J. Sci. Technol. Res, 8(12), 2485-2489.
8. Abdullah, P. Y., Zeebaree, S. R., Shukur, H. M., & Jacksi, K. (2020). HRM system using cloud computing for Small and Medium Enterprises (SMEs). *Technology Reports of Kansai University*, *62*(04), 04.
9. Al-Jaroodi, J., Mohamed, N., & Al Nuaimi, K. (2012). An efficient fault-tolerant algorithm for distributed cloud services. *Proceedings - IEEE 2nd Symposium on Network Cloud Computing and Applications, NCCA 2012*, 1–8. https://doi.org/10.1109/NCCA.2012.21
10. Amoon, M. (2016). Adaptive Framework for Reliable Cloud Computing Environment. *IEEE Access*, *4*, 9469–9478. https://doi.org/10.1109/ACCESS.2016.2623633
11. Araujo Neto, J. P., Pianto, D. M., & Ralha, C. G. (2019). MULTS: A multi-cloud fault-tolerant architecture to manage transient servers in cloud computing. *Journal of Systems Architecture*, *101*. https://doi.org/10.1016/j.sysarc.2019.101651
12. Attallah, S. M. A., Fayek, M. B., Nassar, S. M., & Hemayed, E. E. (2021). Proactive load balancing fault tolerance algorithm in cloud computing. *Concurrency and Computation: Practice and Experience*, *33*(10). https://doi.org/10.1002/cpe.6172

13. Bala, A., & Chana, I. (2012). *Fault Tolerance-Challenges, Techniques and Implementation in Cloud Computing*. www.IJCSI.org
14. Chang, H.-T., Chang, Y.-M., & Hsiao, S.-Y. (2014). Scalable network file systems with load balancing and fault tolerance for web services. *Journal of Systems and Software*, *93*, 102–109. https://doi.org/10.1016/j.jss.2014.02.057
15. Chatterjee, M., Mitra, A., Setua, S. K., & Roy, S. (2020). Gossip-based fault-tolerant load balancing algorithm with low communication overhead. *Computers & Electrical Engineering*, *81*, 106517. https://doi.org/10.1016/j.compeleceng.2019.106517
16. Cotroneo, D., De Simone, L., Liguori, P., & Natella, R. (2023). Run-time failure detection via non-intrusive event analysis in a large-scale cloud computing platform. *Journal of Systems and Software*, *198*. https://doi.org/10.1016/j.jss.2023.111611
17. Dias, A. L., Turcato, A. C., Sestito, G. S., Brandao, D., & Nicoletti, R. (2021). A cloud-based condition monitoring system for fault detection in rotating machines using PROFINET process data. *Computers in Industry*, *126*. https://doi.org/10.1016/j.compind.2021.103394
18. Fang, J., Chao, P., Zhang, R., & Zhou, X. (2019). Integrating workload balancing and fault tolerance in distributed stream processing system. *World Wide Web*, *22*(6), 2471–2496. https://doi.org/10.1007/s11280-018-0656-0
19. Zebari, I. M., Zeebaree, S. R., & Yasin, H. M. (2019, April). Real time video streaming from multi-source using client-server for video distribution. In *2019 4th Scientific International Conference Najaf (SICN)* (pp. 109-114). IEEE.
20. Shukur, H., Zeebaree, S., Zebari, R., Ahmed, O., Haji, L., & Abdulqader, D. (2020). Cache coherence protocols in distributed systems. *Journal of Applied Science and Technology Trends*, *1*(3), 92-97.
21. Sadeeq, M. A., & Zeebaree, S. R. (2023). Design and implementation of an energy management system based on distributed IoT. Computers and Electrical Engineering, 109, 108775.
22. Ibrahem, A. H., & Zeebaree, S. R. (2024). Tackling the Challenges of Distributed Data Management in Cloud Computing-A Review of Approaches and Solutions. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(15s), 340-355
23. Haji, L., Ahmed, O., Sallow, A. B., Haji, L. M., Zeebaree, S. R. M., Ahmed, O. M., Sallow, A. B., Jacksi, K., & Zeabri, R. R. (2020). *Dynamic Resource Allocation for Distributed Systems and Cloud Computing*. https://www.researchgate.net/publication/342317991
24. Hamouda, R. Ben, Hafaiedh, I. Ben, & Robbana, R. (2021). Modelling and verification of reconfigurable fault-tolerant and self-recovering systems in hybrid Clouds. *Simulation Modelling Practice and Theory*, *111*. https://doi.org/10.1016/j.simpat.2021.102331
25. Jain, A., Singh, P., & Jain, E. A. (2014). Survey Paper on Cloud Computing. In *Article in International Journal of Innovations in Engineering and Technology*. https://www.researchgate.net/publication/264435521
26. Jhawar, R., & Piuri, V. (2017). Fault Tolerance and Resilience in Cloud Computing Environments. In *Computer and Information Security Handbook* (pp. 165–181). Elsevier. https://doi.org/10.1016/B978-0-12-803843-7.00009-0
27. Karande, V. M., & Pais, A. R. (n.d.). *CCIS 193 - A Framework for Intrusion Tolerance in Cloud Computing*.
28. Kathpal, C., & Garg, R. (2019). Survey on Fault-Tolerance-Aware Scheduling in Cloud Computing. In *Lecture Notes in Networks and Systems* (Vol. 40, pp. 275–283). Springer. https://doi.org/10.1007/978-981-13-0586-3_28
29. Kumar, V., & Sharma, S. (2016). A Comparative Review on Fault Tolerance methods and models in Cloud Computing. In *International Research Journal of Engineering and Technology*. www.irjet.net
30. Abdullah, H. S., & Zeebaree, S. R. (2024). Distributed Algorithms for Large-Scale Computing in Cloud Environments: A Review of Parallel and Distributed Processing. International Journal of Intelligent Systems and Applications in Engineering, 12(15s), 356-365.
31. Ageed, Z. S., & Zeebaree, S. R. (2024). Distributed Systems Meet Cloud Computing: A Review of Convergence and Integration. International Journal of Intelligent Systems and Applications in Engineering, 12(11s), 469-490.
32. Li, Z., Chang, V., Hu, H., Hu, H., Li, C., & Ge, J. (2021). Real-time and dynamic fault-tolerant scheduling for scientific workflows in clouds. *Information Sciences*, *568*, 13–39. https://doi.org/10.1016/j.ins.2021.03.003
33. Mohammadian, V., Navimipour, N. J., Hosseinzadeh, M., & Darwesh, A. (2022). Fault-Tolerant Load Balancing in Cloud Computing: A Systematic Literature Review. *IEEE Access*, *10*, 12714–12731. https://doi.org/10.1109/ACCESS.2021.3139730
34. Mohammed, B., Kiran, M., Maiyama, K. M., Kamala, M. M., & Awan, I. U. (2017). Failover strategy for fault tolerance in cloud computing environment. *Software - Practice and Experience*, *47*(9), 1243–1274. https://doi.org/10.1002/spe.2491
35. Mukwevho, M. A., & Celik, T. (2021). Toward a Smart Cloud: A Review of Fault-Tolerance Methods in Cloud Systems. *IEEE Transactions on Services Computing*, *14*(2), 589–605. https://doi.org/10.1109/TSC.2018.2816644
36. Nazari Cheraghlou, M., Khadem-Zadeh, A., & Haghparast, M. (2016). A survey of fault tolerance architecture in cloud computing. In *Journal of Network and Computer Applications* (Vol. 61, pp. 81–92). Academic Press. https://doi.org/10.1016/j.jnca.2015.10.004
37. Poola, D., Salehi, M. A., Ramamohanarao, K., & Buyya, R. (2017). A Taxonomy and Survey of Fault-Tolerant Workflow Management Systems in Cloud and Distributed Computing Environments. In *Software Architecture for Big Data and the Cloud* (pp. 285–320). Elsevier. https://doi.org/10.1016/b978-0-12-805467-3.00015-6
38. 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. https://doi.org/10.1016/j.procs.2020.03.106
39. Rahman, Md. M., & Rouf, M. A. (2022). *Aggressive Fault Tolerance in Cloud Computing Using Smart Decision Agent* (pp. 329–344). https://doi.org/10.1007/978-981-16-6636-0_26
40. Rawat, A., Sushil, R., Agarwal, A., Sikander, A., & Bhadoria, R. S. (2023). A New Adaptive Fault Tolerant Framework in the Cloud. *IETE Journal of Research*, *69*(5), 2897–2909. https://doi.org/10.1080/03772063.2021.1907231
41. Rehman, A. U., Aguiar, R. L., & Barraca, J. P. (2022). Fault-Tolerance in the Scope of Cloud Computing. *IEEE Access*, *10*, 63422–63441. https://doi.org/10.1109/ACCESS.2022.3182211
42. Jubair, M. A., Mostafa, S. A., Zebari, D. A., Hariz, H. M., Abdulsattar, N. F., Hassan, M. H., ... & Alouane, M. T. H. (2022). A QoS aware cluster head selection and hybrid cryptography routing protocol for enhancing efficiency and security of VANETs. *IEEE Access*, *10*, 124792-124804.
43. Rezaeipanah, A., Mojarad, M., & Fakhari, A. (2022). Providing a new approach to increase fault tolerance in cloud computing using fuzzy logic. *International Journal of Computers and Applications*, *44*(2), 139–147. https://doi.org/10.1080/1206212X.2019.1709288
44. Rong, H., Liu, J., Wu, W., Hao, J., Wang, H., & Xian, M. (2020). Toward fault-tolerant and secure frequent itemset mining outsourcing in hybrid cloud environment. *Computers and Security*, *98*. https://doi.org/10.1016/j.cose.2020.101969
45. Mohammed, Z. K., Mohammed, M. A., Abdulkareem, K. H., Zebari, D. A., Lakhan, A., Marhoon, H. A., ... & Martinek, R. (2024). A metaverse framework for IoT-based remote patient monitoring and virtual consultations using AES-256 encryption. *Applied Soft Computing*, *158*, 111588.

46. Salih Ageed, Z., R. M. Zeebaree, S., Mohammed Sadeeq, M., Fattah Kak, S., Saeed Yahia, H., R. Mahmood, M., & Mahmood Ibrahim, I. (2021). Comprehensive Survey of Big Data Mining Approaches in Cloud Systems. *Qubahan Academic Journal*, *1*(2), 29–38. https://doi.org/10.48161/qaj.v1n2a46
47. Sathiyamoorthi, V., Keerthika, P., Suresh, P., Zhang, Z., Rao, A. P., & Logeswaran, K. (2021). Adaptive fault tolerant resource allocation scheme for cloud computing environments. *Journal of Organizational and End User Computing*, *33*(5), 1–24. https://doi.org/10.4018/JOEUC.20210901.oa7
48. Saxena, D., & Singh, A. K. (2022). OFP-TM: an online VM failure prediction and tolerance model towards high availability of cloud computing environments. *Journal of Supercomputing*, *78*(6), 8003–8024. https://doi.org/10.1007/s11227-021-04235-z
49. 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. https://doi.org/10.1016/j.jpdc.2019.09.004
50. Shahid, M. A., Islam, N., Alam, M. M., Mazliham, M. S., & Musa, S. (2021). Towards Resilient Method: An exhaustive survey of fault tolerance methods in the cloud computing environment. In *Computer Science Review* (Vol. 40). Elsevier Ireland Ltd. https://doi.org/10.1016/j.cosrev.2021.100398
51. Shahid, M. A., Islam, N., Alam, M. M., Su'Ud, M. M., & Musa, S. (2020). A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach. *IEEE Access*, *8*, 130500–130526. https://doi.org/10.1109/ACCESS.2020.3009184
52. Singh, S., Jeong, Y.-S., & Park, J. H. (2016). A survey on cloud computing security: Issues, threats, and solutions. *Journal of Network and Computer Applications*, *75*, 200–222. https://doi.org/10.1016/j.jnca.2016.09.002
53. Tang, X. (2022). Reliability-Aware Cost-Efficient Scientific Workflows Scheduling Strategy on Multi-Cloud Systems. *IEEE Transactions on Cloud Computing*, *10*(4), 2909–2919. https://doi.org/10.1109/TCC.2021.3057422
54. Zankoya Zaxo, Duhok Polytechnic University, IEEE Computational Intelligence Society. Iraq Chapter., IEEE Communications Society. Iraq Chapter., & Institute of Electrical and Electronics Engineers. (2018). *Distributed Cloud Computing and Distributed Parallel Computing: A Review.*
55. Zhang, P., Chen, Y., Zhou, M., Xu, G., Huang, W., Al-Turki, Y., & Abusorrah, A. (2022). A Fault-Tolerant Model for Performance Optimization of a Fog Computing System. *IEEE Internet of Things Journal*, *9*(3), 1725–1736. https://doi.org/10.1109/JIOT.2021.3088417
56. Zhuang, S., Li, Z., Zhuo, D., Wang, S., Liang, E., Nishihara, R., Moritz, P., & Stoica, I. (2021). Hoplite: Efficient and fault-tolerant collective communication for task-based distributed systems. *SIGCOMM 2021 - Proceedings of the ACM SIGCOMM 2021 Conference*, 641–656. https://doi.org/10.1145/3452296.3472897