Programming and Algorithm Fundamentals
E-ISSN: 3123-979X

Research Article

Complexity Analysis of Adaptive Scheduling Algorithms for Real Time Parallel Processing in Cloud Computing Platforms with Fault Tolerance Mechanisms

Warto [1], Iif Alfiatul Mukaromah [2], and Firstname Lastname [2,*]

1 Universitas Islam Negeri Prof. K.H. Saifuddin Zuhri warto@uinsaizu.ac.id
2 Universitas Islam Negeri Prof. K.H. Saifuddin Zuhri iifam1604@gmail.com
* Corresponding Author : warto@uinsaizu.ac.id

Abstract: The increasing demand for real time parallel processing in cloud computing environments necessitates the development of more efficient and fault-tolerant scheduling algorithms. Traditional scheduling methods, such as static algorithms, often fall short when handling dynamic workloads and system failures, leading to increased task latency and reduced system performance. In contrast, adaptive scheduling algorithms dynamically adjust to changes in system conditions and workloads, ensuring timely task completion and optimized resource utilization. This study evaluates the performance of adaptive scheduling algorithms in real time cloud environments, focusing on key factors such as task latency, system resilience, and fault tolerance. Simulation experiments were conducted using cloud computing models that incorporate fault injection scenarios, including network failures and virtual machine crashes. The results show that adaptive algorithms significantly outperform traditional static schedulers in terms of task latency reduction and improved system resilience. These algorithms demonstrated better fault recovery times and ensured consistent real time performance, even under failure conditions. The findings highlight the advantages of adaptive scheduling in cloud environments, particularly for applications requiring rapid data processing and high system reliability. Despite the promising results, challenges remain regarding the scalability and complexity of these algorithms in large-scale cloud systems. Further research is needed to optimize adaptive scheduling algorithms for efficiency, scalability, and comprehensive performance evaluation, taking into account factors such as energy consumption, cost, and reliability. This research contributes to advancing cloud computing infrastructures that can dynamically handle real time tasks and maintain high performance under varying workloads and failures.

Keywords: real time processing; cloud computing; adaptive scheduling; fault tolerance; task latency.

Received: November 20, 2025
Revised: Desember 30, 2025
Accepted: January 14, 2026
Published: January 18, 2026
Curr. Ver.: January 20, 2026

1. Introduction

Real time parallel processing in cloud computing has emerged as a fundamental technique to handle complex computational tasks, particularly in environments where rapid decision-making and high computational throughput are required. By leveraging multiple processors to perform tasks simultaneously, this approach significantly enhances the overall efficiency and performance of computing systems [1]. Real time processing has become increasingly critical for applications such as artificial intelligence, image processing, distributed cybersecurity monitoring, and large-scale data analytics, where immediate data interpretation and response are necessary to maintain system stability and service continuity [2]. In contemporary digital infrastructures, the growing volume of streaming data and distributed workloads has created an urgent need for scalable parallel architectures capable of processing information with minimal latency. Recent studies highlight that real time distributed processing frameworks are also essential in cloud–edge environments, especially for security monitoring, automated incident response, and intelligent network management

systems [3], [4]. The ability of parallel computing to distribute workloads dynamically across cloud nodes reduces execution time, improves system responsiveness, and supports adaptive resource allocation in modern cloud-native infrastructures [5]. Considering the rapid expansion of real time digital services, the investigation of parallel processing paradigms becomes increasingly important as a current critical technological trend whose long-term architectural implications and scalability limits remain insufficiently understood, making it an urgent research domain within distributed cloud systems [3].

However, despite the advantages of real time parallel processing, cloud computing systems continue to face significant challenges, particularly regarding scheduling efficiency and fault tolerance when operating under highly dynamic workloads. Existing studies indicate that many traditional scheduling algorithms are designed for relatively stable environments and therefore struggle to adapt to rapid variations in system load and resource availability, leading to performance inefficiencies and increased processing latency [6]. Furthermore, fault tolerance remains a persistent challenge due to the large-scale, distributed, and heterogeneous nature of cloud infrastructures. Conventional fault-tolerant techniques such as replication and checkpointing are widely implemented; however, these approaches often introduce additional computational overhead, increased storage consumption, and resource redundancy that may degrade overall system performance [7]. Previous research has also explored energy-aware and load-balancing scheduling strategies to improve resource utilization, yet many of these approaches still rely on static or semi-adaptive mechanisms that cannot fully accommodate rapidly fluctuating workloads in modern cloud environments [8], [9]. More recent studies in cloud-native and distributed security architectures highlight the importance of resilient system design and adaptive resource management; however, the integration of intelligent scheduling mechanisms with dynamic fault-tolerance frameworks remains insufficiently explored in the literature [5], [10]. Consequently, there is still limited understanding of how scheduling efficiency, system resilience, and adaptive fault-tolerance strategies can be jointly optimized in real time parallel cloud computing environments, indicating a critical research gap that motivates further investigation into more adaptive and intelligent scheduling frameworks.

Adaptive scheduling algorithms have emerged as a promising approach to improve scheduling efficiency and ensure fault tolerance in cloud computing environments. These algorithms dynamically adjust task priorities and resource allocations based on real time system conditions, workload variations, and potential failure scenarios, enabling more responsive and flexible resource management compared to traditional static scheduling mechanisms [11]. Previous studies have shown that adaptive scheduling mechanisms can significantly reduce task latency and enhance overall system performance by dynamically reacting to workload fluctuations and system failures [12]. In addition, integrating fault-tolerant scheduling strategies with dynamic load balancing mechanisms has been demonstrated to improve system stability and reliability in large-scale cloud infrastructures [9]. However, despite these advancements, many existing approaches still focus on isolated aspects of scheduling, such as load balancing or fault tolerance, rather than addressing the combined optimization of adaptive scheduling, system resilience, and dynamic workload management in real time cloud environments. Recent research in cloud-native system resilience and distributed security infrastructures highlights the growing need for adaptive orchestration mechanisms capable of maintaining service continuity under dynamic operational conditions [13], [14]. Therefore, this study aims to investigate how adaptive scheduling strategies can simultaneously improve scheduling efficiency, fault tolerance, and resource utilization in real time parallel cloud computing environments. Specifically, this article seeks to answer the fundamental research question: how can adaptive scheduling mechanisms be designed to maintain optimal performance and system reliability in cloud computing systems operating under highly dynamic workloads and potential failure conditions?

This study aims to analyze the computational complexity and performance of adaptive scheduling algorithms for fault-tolerant real time cloud systems. Previous research has proposed various fault-tolerant scheduling approaches to improve system reliability and reduce task failure in distributed cloud environments; however, many studies primarily focus on isolated performance metrics such as execution time or resource redundancy rather than providing a comprehensive evaluation that simultaneously considers scheduling efficiency, system resilience, and resource utilization [7]. In contrast, this study investigates the computational characteristics and operational performance of adaptive scheduling strategies by examining models such as Priority-Based Task Scheduling with Fault Tolerance (PBTS-FT) and Dynamic Fault-Tolerant Workflow Scheduling (DFTWS). By integrating

performance evaluation with computational complexity analysis, this research provides a more holistic understanding of how adaptive scheduling mechanisms operate in real time cloud environments. Furthermore, recent studies on cloud-native system resilience emphasize the need for adaptive orchestration and robust architectural frameworks to maintain service continuity under dynamic workloads and failure conditions [5]. Similarly, emerging research in distributed cloud security and real time network monitoring highlights the importance of adaptive system mechanisms that can maintain operational stability in highly dynamic infrastructures [4], [15]. Therefore, the distinctive contribution of this research lies in its integrated evaluation framework that jointly examines computational complexity, scheduling performance, and fault-tolerance capability, providing deeper insights into how adaptive scheduling algorithms can optimize execution efficiency, minimize task latency, and enhance system resilience in real time cloud computing environments.

2. Literature Review

Real Time Processing in Cloud Computing

Real time processing in cloud computing environments refers to the capability of distributed computing infrastructures to process data streams and computational tasks with minimal latency while maintaining system responsiveness. As modern cloud systems increasingly support data-intensive applications such as Internet of Things (IoT), intelligent monitoring systems, and distributed analytics, the need for efficient real time data processing mechanisms has become more critical. Traditional cloud computing architectures often experience latency issues due to transmission bottlenecks, centralized processing models, and limited resource allocation strategies, which can significantly affect the performance of time sensitive applications [16]. To address these limitations, recent research has explored the integration of fog and edge computing architectures that enable data processing closer to the data source. Such distributed architectures reduce communication delays, enhance system scalability, and improve processing performance in real time environments [17]. For example, scalable cloud-based real time data processing frameworks have demonstrated improved efficiency in industrial and manufacturing environments by optimizing data pipelines and computational workflows [18]. Furthermore, modern cloud-native architectures emphasize resilience and distributed resource orchestration as key factors in enabling reliable real time processing in large-scale cloud systems [5]. Therefore, real time processing in cloud environments can be conceptualized as a distributed computational paradigm that combines scalable cloud infrastructure, edge/fog computing integration, and adaptive resource management to support latency-sensitive applications.

From a research perspective, the concept of real time processing in cloud computing can be operationalized through several measurable variables related to system performance and operational efficiency. One important variable is processing latency, which measures the time required for a system to receive, process, and respond to incoming data streams in real time environments. Another critical variable is processing throughput, representing the volume of data or tasks that can be processed within a given time interval, reflecting the scalability of cloud computing infrastructures [16]. In addition, system resource utilization is commonly used to evaluate how efficiently computing resources such as CPU, memory, and network bandwidth are allocated and utilized during real time task execution. Distributed processing architectures such as fog computing and edge-based frameworks can significantly improve these performance variables by reducing network delays and optimizing computational distribution [17]. Moreover, system resilience and reliability have become increasingly important variables in modern cloud-native environments, as real time processing systems must maintain stable operation even under fluctuating workloads and infrastructure disruptions [5]. In practical implementations such as real time monitoring systems and distributed IoT environments, efficient real time processing frameworks enable faster data interpretation and adaptive system responses, thereby supporting intelligent decision-making in dynamic digital ecosystems [19]. Consequently, these variables provide a comprehensive framework for evaluating the effectiveness of real time processing mechanisms in cloud computing environments.

Scheduling Algorithms in Cloud Computing

Scheduling algorithms play a critical role in cloud computing environments because they determine how computational tasks are allocated to available resources such as virtual machines, processors, and storage nodes. In general, scheduling algorithms in cloud computing can be classified into two main categories: static scheduling and adaptive scheduling. Static scheduling algorithms, including First-Come-First-Serve (FCFS) and Shortest Job First (SJF), allocate tasks based on predetermined rules without considering dynamic system conditions or workload fluctuations. These algorithms are relatively simple to implement and efficient in stable environments where workloads remain predictable [20]. However, in modern cloud infrastructures characterized by dynamic workloads, heterogeneous resources, and varying network conditions, static approaches often struggle to maintain optimal performance and may lead to inefficient resource utilization [21]. To address these limitations, adaptive scheduling algorithms have been proposed to dynamically adjust task allocation based on real time system conditions. For example, Dynamic Priority Task-Based Scheduling (DPTS) modifies task priorities according to current system performance, while Adaptive Workflow Scheduling-CTDC (AWS-CTDC) adapts scheduling decisions based on virtual machine performance and deadline constraints [22], [23]. Such adaptive mechanisms are particularly important in cloud-native systems where dynamic resource orchestration and resilient system architectures are required to maintain service continuity [5].

From an analytical perspective, the effectiveness of scheduling algorithms in cloud computing can be evaluated through several measurable performance variables. One fundamental variable is task scheduling efficiency, which reflects how effectively tasks are assigned to available computing resources while minimizing waiting time and processing delays. Another important variable is system throughput, which measures the number of tasks completed within a specific time period and indicates the scalability of the scheduling mechanism [20]. In addition, task latency is commonly used to evaluate the responsiveness of scheduling algorithms, particularly in real time cloud environments where tasks must be completed within strict deadlines. Adaptive scheduling algorithms typically aim to reduce latency by dynamically adjusting resource allocation and task priorities in response to workload variations [24]. Furthermore, resource utilization represents how efficiently computing resources such as CPU, memory, and virtual machines are used during task execution. Efficient scheduling mechanisms help prevent resource underutilization and bottlenecks in distributed systems [21]. In modern cloud environments, system resilience and reliability also serve as important variables because scheduling algorithms must maintain system stability even in the presence of failures, network fluctuations, or workload spikes [5]. These variables collectively provide a framework for evaluating the performance and adaptability of scheduling algorithms in dynamic cloud computing infrastructures.

Fault Tolerance Mechanisms in Cloud Computing

Fault tolerance is a fundamental concept in cloud computing systems because it ensures continuous service availability even when system components experience failures. In large-scale distributed cloud environments, failures may occur due to hardware malfunctions, software errors, network disruptions, or sudden workload spikes. Therefore, fault tolerance mechanisms are implemented to maintain system reliability and prevent service interruptions. One commonly used mechanism is task replication, where multiple copies of a task are executed across different computing nodes to ensure that at least one instance completes successfully in the event of a failure [25]. Another widely used approach is checkpointing, which periodically stores the execution state of a task so that it can resume from the most recent checkpoint if a failure occurs, thereby reducing the need to restart the entire computation process [26]. In addition, redundancy and load balancing mechanisms distribute tasks across multiple resources to prevent overload and ensure system stability in distributed environments [27]. These mechanisms collectively contribute to improving the reliability and availability of cloud systems, although they often introduce additional computational overhead and resource consumption. In modern cloud-native infrastructures, system resilience and architectural robustness have become increasingly important design considerations to ensure that fault tolerance mechanisms operate efficiently under dynamic workloads [5].

From an analytical standpoint, the effectiveness of fault tolerance mechanisms in cloud computing can be evaluated through several operational variables related to system performance and reliability. One key variable is system reliability, which measures the probability that a cloud system continues to operate correctly without interruption despite potential component failures. Another important variable is recovery time, which represents the duration required for the system to restore its normal operation after a failure occurs. Techniques such as checkpointing significantly reduce recovery time by enabling tasks to resume from previously saved states instead of restarting the entire execution process [26]. In addition, resource overhead is an essential variable because many fault tolerance mechanisms, including replication and redundancy strategies, require additional computational resources and storage capacity to maintain system reliability [25]. Another relevant variable is load distribution efficiency, which evaluates how effectively system workloads are balanced across distributed resources to prevent bottlenecks and system failures [27]. Furthermore, modern cloud infrastructures emphasize system resilience, which reflects the ability of distributed architectures to maintain service continuity even under unpredictable failures and dynamic operational conditions [5], [28]. These variables collectively provide a framework for evaluating the performance and effectiveness of fault tolerance strategies in cloud computing environments.

Adaptive Scheduling for Fault-Tolerant Real Time Cloud Systems

Adaptive scheduling for fault-tolerant real time cloud systems can be conceptualized as an intelligent task management approach that dynamically adjusts scheduling decisions in response to changing workloads, timing constraints, resource availability, and system failure conditions. In cloud computing environments, static scheduling methods are often insufficient because they allocate tasks based on predefined assumptions and cannot effectively respond to runtime changes, such as node failure, workload spikes, or performance degradation [29], [30]. By contrast, adaptive scheduling mechanisms are designed to continuously evaluate current system states and modify task assignment strategies accordingly in order to preserve execution efficiency and deadline compliance. In real time environments, this adaptability becomes even more critical because the system must not only complete tasks correctly but also ensure that they are executed within strict timing requirements. Recent studies have shown that adaptive fault-tolerant algorithms can improve both reliability and timing performance in heterogeneous systems by incorporating transient fault handling and timing-aware task allocation strategies [31]. In cloud-native infrastructures, this concept is further strengthened by the growing importance of resilient software architecture, where scheduling intelligence must be integrated with system robustness to maintain service continuity under dynamic conditions [5]. Therefore, adaptive scheduling in fault-tolerant real time cloud systems may be understood as a dynamic orchestration mechanism that combines scheduling flexibility, timing awareness, and fault recovery capability to support continuous and reliable cloud service execution.

From an empirical perspective, adaptive scheduling for fault-tolerant real time cloud systems can be operationalized through several measurable variables that reflect both performance and resilience. One primary variable is task completion time, which measures how quickly tasks are executed under adaptive scheduling policies, especially in time-sensitive computing environments. Another important variable is deadline satisfaction, which indicates the extent to which scheduled tasks are completed within predefined real time constraints; this is particularly relevant because adaptive scheduling is intended to preserve timeliness under fluctuating workloads and transient failures [31]. A further variable is fault recovery effectiveness, which reflects the system's ability to maintain operation or rapidly recover when failures occur. In addition, resource utilization efficiency is an essential variable because adaptive scheduling should allocate computing resources in a manner that minimizes waste while preserving system performance [28]. Another relevant indicator is energy efficiency, since cloud task scheduling strategies also influence power consumption, especially when comparing dynamic and static scheduling approaches in large-scale infrastructures [29]. Finally, system resilience can be used as a higher-level variable to evaluate the capacity of cloud-native systems to sustain reliable operation despite workload volatility and infrastructure disruption [5]. Collectively, these variables provide a comprehensive analytical framework for assessing the effectiveness of adaptive scheduling in fault-tolerant real time cloud environments.

3. Research Method

The research employs a computational complexity analysis approach to evaluate the performance of adaptive scheduling algorithms in fault-tolerant real time cloud systems. Using simulation-based experiments with cloud computing models, multiple virtual machines (VMs), and fault injection scenarios, the study assesses key parameters such as task latency and system resilience. The simulations simulate network failures and server crashes to test the algorithms' ability to recover and maintain real time task execution. Tools like CloudSim, GreenCloud, and SimGrid are used for modeling cloud environments and fault conditions, providing a comprehensive evaluation of the algorithms' efficiency, scalability, and fault tolerance in dynamic cloud environments.

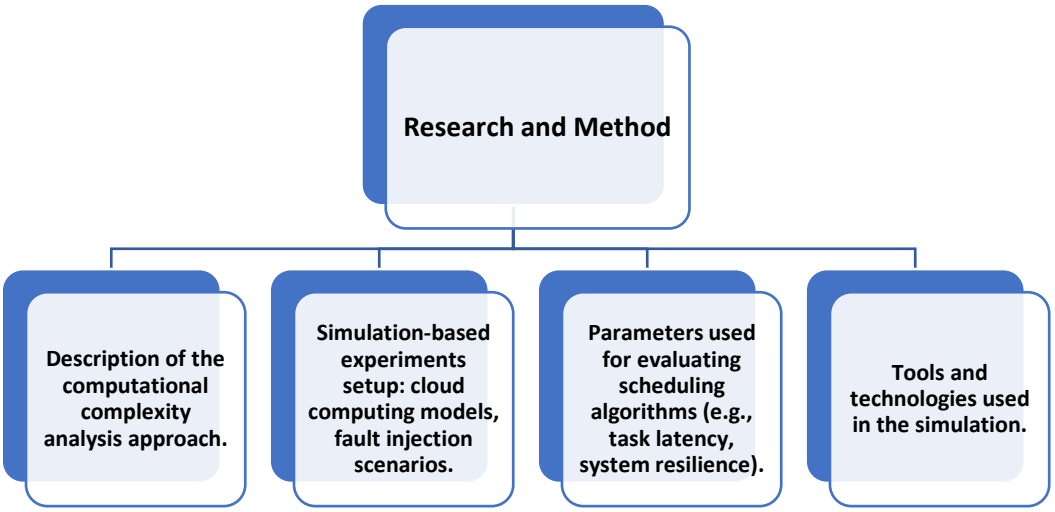

Figure 1. Flowchart structure.

Description of the Computational Complexity Analysis Approach

To assess the performance of adaptive scheduling algorithms in fault-tolerant real time cloud systems, this research employs a computational complexity analysis approach. This analysis focuses on evaluating how scheduling algorithms perform under varying system loads, resource availability, and failure scenarios. The complexity analysis evaluates task execution time, resource utilization, and system responsiveness to real time conditions. It is aimed at identifying the trade-offs between scheduling efficiency and fault tolerance, especially in dynamic cloud environments.

Simulation-Based Experiments Setup: Cloud Computing Models, Fault Injection Scenarios

The research utilizes a simulation-based approach to evaluate the effectiveness of adaptive scheduling algorithms in real-world cloud environments. The cloud computing models used in the simulations are designed to mimic real time workloads and fault tolerance mechanisms in large-scale systems. The cloud models integrate multiple virtual machines (VMs), each representing an independent processing node capable of executing parallel tasks. Fault injection scenarios are incorporated to test the resilience of the scheduling algorithms under failure conditions. These fault injection scenarios simulate network failures, server crashes, and unexpected resource unavailability, which are common challenges in cloud computing. The simulations help evaluate how well the algorithms can recover from failures while maintaining real time task execution.

Parameters Used for Evaluating Scheduling Algorithms

The effectiveness of the adaptive scheduling algorithms is evaluated using several key parameters. First, task latency is measured, which refers to the time taken from the task submission to its completion. Minimizing task latency is crucial for ensuring real time performance, particularly in applications where delays are unacceptable. Second, system resilience is assessed by evaluating how well the system continues to operate under failure

conditions, which is directly related to the fault tolerance mechanisms incorporated into the scheduling algorithms. Key factors for resilience include recovery time, the impact of failures on task completion, and system throughput during failures. These parameters are essential for understanding the trade-offs between scheduling efficiency and fault tolerance in cloud environments.

Tools and Technologies Used in the Simulation

The simulation is carried out using cloud computing simulation tools such as CloudSim and GreenCloud. CloudSim, a widely used simulation toolkit for cloud computing, enables the modeling of various cloud environments, including task scheduling, resource allocation, and fault tolerance mechanisms. GreenCloud is another tool that is particularly useful for simulating large-scale data centers and evaluating energy-efficient cloud scheduling algorithms. The simulations also use fault injection tools to simulate failure conditions, such as SimGrid, which provides a framework for testing distributed applications under network and hardware failure scenarios. These tools enable a realistic evaluation of the adaptive scheduling algorithms, considering various cloud configurations, fault tolerance techniques, and performance metrics.

4. Results and Discussion

The simulation results showed that adaptive scheduling algorithms significantly outperform traditional static schedulers in real time cloud computing environments. Adaptive algorithms, such as Dynamic Priority Task-Based Scheduling (DPTS) and Adaptive Workflow Scheduling-CTDC (AWS-CTDC), effectively reduced task latency by dynamically adjusting resource allocation and task priorities, ensuring efficient task execution under varying workloads. They also demonstrated better system resilience by quickly recovering from fault scenarios like network failures or virtual machine crashes, minimizing downtime and maintaining real time performance. In contrast, static algorithms, which follow fixed schedules, struggled to adapt to dynamic changes, leading to higher latency and slower recovery during failures. While adaptive scheduling offers superior performance, its complexity and computational overhead remain challenges, especially in large-scale cloud environments, necessitating further optimization to enhance scalability and efficiency.

Results

The simulation experiments demonstrated the significant advantages of adaptive scheduling algorithms over traditional static scheduling methods in real time cloud computing environments. Adaptive scheduling algorithms, such as Dynamic Priority Task-Based Scheduling (DPTS) and Adaptive Workflow Scheduling-CTDC (AWS-CTDC), significantly reduced task latency compared to static algorithms like First-Come-First-Serve (FCFS) and Shortest Job First (SJF). Task latency, defined as the time taken from task submission to completion, was consistently lower in the adaptive scheduling scenarios, especially during periods of high system load. This improvement was due to the dynamic allocation of resources and task priorities, which allowed for efficient task execution even under fluctuating workload conditions. Furthermore, the adaptive scheduling algorithms demonstrated enhanced system resilience when fault scenarios were introduced, such as network failures and virtual machine crashes. The system recovery times were shorter, and the impact of failures on task completion was minimized.

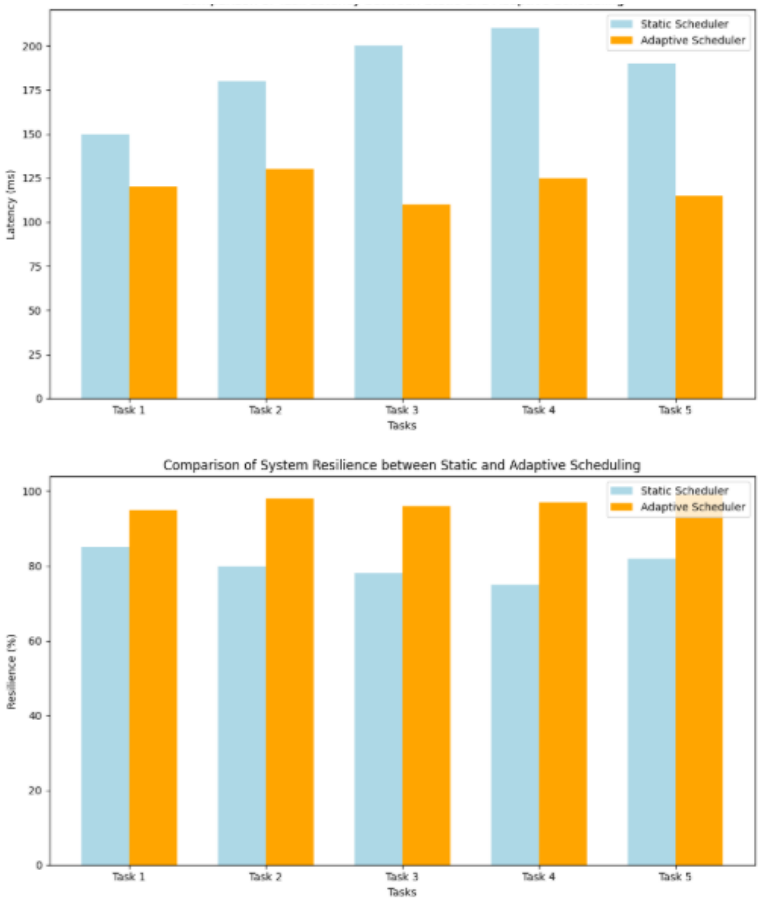

Figure 2,3 Comparison of Task Latency between Static and Adaptive Scheduling, Comparison of System Resilience between Static and Adaptive Scheduling.

The results are supported by graphical analysis comparing the performance of static and adaptive scheduling algorithms in cloud computing environments. The task latency comparison shows that adaptive scheduling consistently achieves lower latency across all evaluated tasks, indicating faster task execution and improved responsiveness compared to static scheduling methods. This improvement occurs because adaptive schedulers dynamically adjust resource allocation based on current system conditions. In addition, the system resilience comparison demonstrates that adaptive scheduling achieves a higher percentage of successfully completed tasks, even under failure conditions. These findings indicate that adaptive scheduling mechanisms are more effective in handling dynamic workloads while maintaining system reliability and performance.

In contrast, static scheduling algorithms were less efficient in handling dynamic changes in system load and failure conditions. These algorithms, which follow fixed task schedules, were unable to adjust to changes in resource availability or workload fluctuations. As a result, task latency increased, and the system struggled to maintain real time performance under fault scenarios. Static schedulers also exhibited slower recovery times and higher failure rates when virtual machines crashed or resources became unavailable. While they may be simpler to implement, static scheduling algorithms lack the flexibility and adaptability required to maintain system performance in complex cloud environments.

Discussion

The results highlight the superior performance of adaptive scheduling algorithms in real time cloud computing systems. The reduction in task latency observed with adaptive scheduling algorithms is crucial for applications that demand timely task completion, such as artificial intelligence, image processing, and large-scale data analysis. Adaptive scheduling, by dynamically adjusting task priorities and resource allocations, ensures that high-priority tasks are executed first, reducing delays and improving system responsiveness. This adaptability allows adaptive algorithms to meet real time processing requirements more efficiently than

static schedulers, which are limited by their inability to respond to workload changes. The ability to minimize task latency in dynamic environments is a key advantage that makes adaptive scheduling a promising approach for real time cloud systems.

In addition to task latency, the improved system resilience observed with adaptive scheduling algorithms further underscores their effectiveness in cloud computing environments. The fault tolerance mechanisms integrated into these algorithms, such as task replication and checkpointing, were more efficiently managed during failure scenarios, ensuring that the system continued to operate smoothly even when faced with network disruptions or virtual machine crashes. The dynamic nature of adaptive algorithms allowed for quick reallocation of tasks to available resources, ensuring that task deadlines were met and system performance remained consistent. In contrast, static scheduling algorithms struggled to recover from failures, leading to longer recovery times and more missed deadlines. This highlights the critical importance of adaptive scheduling in ensuring the reliability and availability of cloud systems, particularly when dealing with real time applications.

However, while the results are promising, there are still challenges associated with the implementation of adaptive scheduling algorithms. One of the key issues is the complexity of these algorithms compared to static schedulers. Adaptive algorithms require more sophisticated mechanisms to monitor system conditions, adjust task priorities, and manage fault tolerance effectively. This complexity can lead to higher computational overhead, which may reduce the overall efficiency of the system in certain scenarios. Additionally, the scalability of adaptive algorithms remains a concern, particularly in large-scale cloud environments with millions of tasks and resources. Future research should focus on optimizing these algorithms to reduce overhead and improve scalability, ensuring that they can handle the increasing demands of large cloud systems while maintaining low latency and high resilience.

5. Comparison

The comparison between adaptive and traditional scheduling algorithms reveals several key differences in terms of performance, flexibility, and fault tolerance. Traditional scheduling algorithms, such as First-Come-First-Serve (FCFS) and Shortest Job First (SJF), follow predefined task sequences without considering real time changes in system load or resource availability. These static algorithms, while straightforward and efficient in stable environments, fail to adapt when the system experiences fluctuations in workload or resource failure. As a result, static schedulers often suffer from inefficiencies, including increased task latency and a higher likelihood of missed deadlines under dynamic conditions. In contrast, adaptive scheduling algorithms, such as Dynamic Priority Task-Based Scheduling (DPTS) and Adaptive Workflow Scheduling-CTDC (AWS-CTDC), are designed to dynamically adjust task priorities and resource allocations in response to real time changes in workload and system state. This flexibility allows adaptive algorithms to optimize resource utilization, reduce task latency, and ensure timely task completion, even under fluctuating conditions.

One of the main advantages of adaptive scheduling algorithms is their ability to dynamically adjust to workload changes and system failures. While static scheduling methods are unable to respond to system changes, adaptive algorithms continuously monitor the environment and reallocate resources based on current conditions. For instance, in scenarios where system resources become overburdened or tasks need to be prioritized differently, adaptive algorithms can adjust in real time to ensure that critical tasks are completed on time. In contrast, static schedulers do not have this capability and often continue executing tasks based on their initial schedules, which can lead to delays and inefficiencies. Additionally, adaptive algorithms are better equipped to handle fault scenarios, such as server crashes or network failures, by reassigning tasks to available resources and minimizing the impact of the failure on task completion. Static scheduling algorithms, however, struggle to recover from such failures, often leading to longer recovery times and missed deadlines.

Fault tolerance is another area where adaptive scheduling algorithms outperform traditional scheduling methods. Adaptive algorithms integrate fault tolerance mechanisms, such as task replication and checkpointing, which allow the system to continue operating seamlessly in the event of a failure. These algorithms can dynamically adjust to failures by replicating tasks across multiple resources or resuming tasks from the last checkpoint, thus minimizing downtime and ensuring that critical tasks are completed. In comparison, static

scheduling algorithms are not equipped to handle failures effectively, as they lack the flexibility to reassign tasks or adjust priorities dynamically. The failure to respond to system issues in real time leads to increased resource wastage, prolonged task delays, and decreased system reliability. Therefore, adaptive scheduling algorithms offer significant fault tolerance benefits, improving the overall resilience of the system and ensuring continuous real time performance, even under adverse conditions.

In summary, adaptive scheduling algorithms provide distinct advantages over traditional static scheduling methods by dynamically adjusting to changes in workload and system conditions. Their ability to handle failure scenarios, prioritize tasks effectively, and optimize resource utilization results in lower task latency, improved system resilience, and enhanced real time performance. These benefits make adaptive scheduling algorithms particularly well-suited for cloud environments, where workloads are dynamic, and maintaining high performance and fault tolerance is essential. While static algorithms may still be suitable for simpler, stable environments, adaptive scheduling is the preferred approach for complex, real time cloud computing applications.

6. Conclusions

In summary, the findings from the simulation experiments clearly demonstrate the advantages of adaptive scheduling algorithms over traditional static scheduling methods in real time cloud computing environments. Adaptive scheduling algorithms significantly reduced task latency and improved system resilience, especially under dynamic workloads and fault injection scenarios. The ability of adaptive algorithms to adjust task priorities and resource allocations in real time allowed for more efficient resource utilization, minimized task delays, and ensured that system performance remained optimal, even in the face of failures. These improvements in task latency and system resilience highlight the potential of adaptive scheduling algorithms to enhance the overall performance and reliability of cloud systems.

The implications for real time parallel processing in cloud computing environments are significant. The reduction in task latency and the enhancement of system resilience make adaptive scheduling algorithms an ideal choice for cloud applications that require rapid data processing, such as artificial intelligence, image processing, and large-scale data analysis. As cloud environments continue to grow in scale and complexity, the ability to dynamically adjust scheduling and fault tolerance mechanisms will be critical in maintaining high system performance and reliability. Adaptive scheduling ensures that real time tasks are completed on time and that the system can recover quickly from failures, making it a vital component of modern cloud computing infrastructures.

Further research in adaptive scheduling algorithms and fault tolerance mechanisms is essential to address existing challenges and optimize their performance in large-scale cloud environments. Future studies should focus on enhancing the scalability and efficiency of adaptive algorithms, ensuring that they can handle increasing workloads without introducing excessive overhead. Additionally, comprehensive evaluation frameworks that consider factors such as energy consumption, cost, and system reliability should be developed to better assess the performance of these algorithms. By advancing the field of adaptive scheduling, researchers can help create more robust, efficient, and fault-tolerant cloud computing systems capable of meeting the growing demands of real time applications.

References

[1] M. Sino and E. Domazet, "Scalable Parallel Processing: Architectural Models, Real-Time Programming, and Performance Evaluation †," *Eng. Proc.*, vol. 104, no. 1, 2025, doi: 10.3390/engproc2025104060.

[2] S. S. Thomas, S. A. Thomas, J. Paul, and K. K. B. Shibu, "An Intelligent Adaptive Scheduler for Operating Systems Experimented Using FreeRTOS," in *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICICCS 2018*, 2018, pp. 1592 – 1597. doi: 10.1109/ICCONS.2018.8662927.

[3] Danang, T. Wahyono, I. Sembiring, T. Wellem, and N. H. Dzulkefly, "An Adaptive Framework Integrating ML Blockchain and TEE for Cloud Security," in *Proceeding - 2025 4th International Conference on Creative Communication and Innovative Technology: Empowering Transformative MATURE LEADERSHIP: Harnessing Technological Advancement for Global Sustainability, ICCIT 2025*, 2025. doi:

10.1109/ICCIT65724.2025.11167152.

[4] D. Danang and Z. Mustofa, "Digital Forensics and Automated Incident Response Framework Leveraging Big Data Analytics and Real Time Network Traffic Profiling in Heterogeneous Cyber Environments," *Cyber Secur. Netw. Manag.*, vol. 1, no. 1, pp. 44–45, 2026.

[5] E. Siswanto, D. Danang, I. Kusumaningroem, and I. Akhsani, "Assessing Software Architecture Resilience Using Quantitative Metrics in Cloud Native Application Development Environments," *Indones. J. Infomatics*, vol. 1, no. 1, pp. 11–21, 2026.

[6] S. U. Mushtaq, S. Sheikh, and S. M. Idrees, "Enhanced priority based task scheduling with integrated fault tolerance in distributed systems," *Int. J. Cogn. Comput. Eng.*, vol. 6, pp. 152 – 169, 2025, doi: 10.1016/j.ijcce.2024.12.006.

[7] A. P. Sheetal and K. Ravindranath, "Cost Effective Hybrid Fault Tolerant Scheduling Model for Cloud Computing Environment: Hybrid Fault Tolerant Scheduling," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 6, pp. 416 – 422, 2021, doi: 10.14569/IJACSA.2021.0120646.

[8] K. Ajmera, T. K. Tewari, V. K. Singh, Vikash, and P. K. Upadhyay, "Energy-Aware Dynamic Virtual Machine Scheduling in Cloud Computing: A Survey," in *ACM International Conference Proceeding Series*, 2023, pp. 133 – 141. doi: 10.1145/3607947.3607970.

[9] T. Hagras and G. A. El-Sayed, "A fault-tolerant and load-balancing scheduler for independent tasks on cloud-based virtual machines," *Cluster Comput.*, vol. 29, no. 1, 2026, doi: 10.1007/s10586-025-05857-1.

[10] D. Danang, A. B. Santoso, and M. U. Dewi, "CICA Framework: Harnessing CSR, AI, and Blockchain for Sustainable Digital Culture," *Int. J. Adv. Comput. Sci. \& Appl.*, vol. 16, no. 11, 2025.

[11] K. Tanaka, "Real-time adaptive task scheduling," in *Proceedings of the 2005 International Conference on Embedded Systems and Applications, ESA'05*, 2005, pp. 24 – 30. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-60749110125&partnerID=40&md5=11898a7f75ac7a9cfb086cdd3d8eea95

[12] N. Wu, D. Zuo, and Z. Zhang, "Dynamic fault-tolerant workflow scheduling with hybrid spatial-temporal re-execution in clouds," *Inf.*, vol. 10, no. 5, 2019, doi: 10.3390/info10050169.

[13] D. Danang, F. A. Prasetya, and E. Siswanto, "Design of Intelligent Street Lighting Systems Based on Motion and Ambient Light Sensors," *J. Multidiscip. Res. Technol.*, vol. 2, no. 1, pp. 65–79, 2026.

[14] D. Danang, I. A. Dianta, A. B. Santoso, and S. Kholifah, "Hybrid CNN GRU Framework for Early Detection and Adaptive Mitigation of DDoS Attacks in SDN using Image Based Traffic Analysis," *Int. J. Inf. Eng. Sci.*, vol. 2, no. 2, pp. 66–78, 2025, doi: https://doi.org/10.62951/ijies.v2i2.292.

[15] D. Danang, E. Siswanto, N. D. Setiawan, and P. Wibowo, "Hybrid Zero Trust Container Based Model for Proactive Service Continuity under Intelligent DDoS Attacks in Cloud Environment," *Int. J. Comput. Technol. Sci.*, vol. 2, no. 3, pp. 41–49, 2025, doi: https://doi.org/10.62951/ijcts.v2i3.291.

[16] S. Sahoo, S. Nawaz, S. K. Mishra, and B. Sahoo, "Execution of real time task on cloud environment," in *12th IEEE International Conference Electronics, Energy, Environment, Communication, Computer, Control: (E3-C3), INDICON 2015*, 2016. doi: 10.1109/INDICON.2015.7443778.

[17] S.-J. Yang and W.-L. Lu, "Design a Distributed Fog Computing Scheme to Enhance Processing Performance in Real-Time IoT Applications," *Lect. Notes Data Eng. Commun. Technol.*, vol. 41, pp. 99 – 112, 2020, doi: 10.1007/978-3-030-34986-8_7.

[18] M. R. Kale, S. Labhane, S. E. Manu, P. Mehta, A. Amudha, and A. Gupta, "Scalable and Efficient Real-Time Data Processing in Cloud-Based Manufacturing Systems," in *2024 15th International Conference on Computing Communication and Networking Technologies, ICCCNT 2024*, 2024. doi: 10.1109/ICCCNT61001.2024.10725934.

[19] D. Danang, N. D. Setiawan, and E. Siswanto, "Pemanfaatan Teknologi Internet of Things untuk Monitoring Kualitas Air Sungai di Wilayah Perkotaan," *J. New Trends Sci.*, vol. 2, no. 1, pp. 23–34, 2024.

[20] P. Akilandeswari and H. Srimathi, "Survey and analysis on task scheduling in cloud environment," *Indian J. Sci. Technol.*, vol. 9, no. 37, 2016, doi: 10.17485/ijst/2016/v9i37/102058.

[21] H. Wang and H. Wang, "Survey On Task Scheduling in Cloud Computing Environment," in *ICIIBMS 2022 - 7th International Conference on Intelligent Informatics and Biomedical Sciences*, 2022, pp. 286 – 291. doi: 10.1109/ICIIBMS55689.2022.9971622.

[22] R. S. K. Aakisetti, V. Ganta, P. Yellamma, C. Siram, S. H. Gampa, and K. V Brahma Rao, "Dynamic Priority Scheduling Algorithms for Flexible Task Management in Cloud Computing," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 13s, pp. 246 – 256, 2024, [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85184431242&partnerID=40&md5=609ac627c3fc050f57af59c347d44ace

[23] R. Ghafouri and A. Movaghar, "An adaptive and deadline-constrained workflow scheduling algorithm in infrastructure as a service clouds," *Iran J. Comput. Sci.*, vol. 5, no. 1, pp. 17 – 39, 2022, doi: 10.1007/s42044-021-00082-6.

[24] Y. Liu, J. Liu, Z. Zhu, C. Deng, Z. Ren, and X. Xu, "Adaptive fault-tolerant scheduling in heterogeneous real-time systems," in *Proceedings of the 14th IEEE Conference on Industrial Electronics and Applications, ICIEA 2019*, 2019, pp. 982 – 987. doi: 10.1109/ICIEA.2019.8833833.

[25] K. Sumangali and N. Benny, "Advanced cloud fault tolerance system," in *IOP Conference Series: Materials Science and Engineering*, 2017. doi: 10.1088/1757-899X/263/4/042060.

[26] P. Marcotte, F. Gregoire, and F. Petrillo, "Multiple fault-Tolerance mechanisms in cloud systems: A systematic review," in *Proceedings - 2019 IEEE 30th International Symposium on Software Reliability Engineering Workshops, ISSREW 2019*, 2019, pp. 414 – 421. doi: 10.1109/ISSREW.2019.00104.

[27] J. Hao, Z. Cui, and Z. Peng, "Load balancing for data centre: A brief survey," *Int. J. Wirel. Mob. Comput.*, vol. 11, no. 1, pp. 47 – 53, 2016, doi: 10.1504/IJWMC.2016.079464.

[28] K. A. Ali, O. A. Fadare, and F. Al-Turjman, "Dynamic Resource Allocation (DRA) in Cloud Computing," *Sustain. Civ. Infrastructures*, vol. Part F4042, pp. 1033 – 1049, 2025, doi: 10.1007/978-3-031-72509-8_85.

[29] A. Khiat, "Optimizing Cloud Energy Consumption Using Static Task Scheduling Algorithms: A Comparative Study," in *2023 14th International Conference on Information and Communication Systems, ICICS 2023*, 2023. doi: 10.1109/ICICS60529.2023.10330466.

[30] O. Gokalp, "Performance evaluation of heuristic and metaheuristic algorithms for independent and static task scheduling in cloud computing; [Bulut hesaplamada bağimsiz ve statik görev çizelgeleme için sezgisel ve metasezgisel algoritmalarin performans değerlendirmesi]," in *SIU 2021 - 29th IEEE Conference on Signal Processing and Communications Applications, Proceedings*, 2021. doi: 10.1109/SIU53274.2021.9477821.

[31] J. Liu, Z. Zhu, and C. Deng, "A Novel and Adaptive Transient Fault-Tolerant Algorithm Considering Timing Constraint on Heterogeneous Systems," *IEEE Access*, vol. 8, pp. 103047 – 103061, 2020, doi: 10.1109/ACCESS.2020.2999092.