mathematics

MDPI

Article

Dynamic and Stochastic Models for Application Management in Distributed Computing Systems

Saleh M. Altowaijri

Department of Information Systems, Faculty of Computing and Information Technology, Northern Border University, Rafha 91911, Saudi Arabia; saleh.altowaijri@nbu.edu.sa

Abstract: Fog and edge computing have proven indispensable in tackling issues related to time-critical applications, high network congestion, user confidentiality, and data protection. While these emerging paradigms offer significant potential, substantial effort is required to study and design systems and applications tailored to their unique characteristics. This study conducts a comprehensive analysis of distributed application scheduling and offloading across cloud, fog, and edge environments. We developed multiple prototypes to investigate the organization of distributed applications under various system scales and workloads. To evaluate the system's effectiveness and reliability, we computed steady-state probabilities using enhanced Markov models specifically designed for cloud, fog, and edge settings. These probabilities were employed to establish key metrics for assessing the efficiency of distributed application scheduling and offloading, including network utilization, response delay, energy consumption, and associated costs.

Keywords: Markov models; stochastic systems; resource allocation; mathematical modeling; distributed applications; hybrid computing systems; cloud; fog; edge computing

MSC: 68M14; 68M20; 60K20

check for updates

Academic Editor: Jonathan Blackledge

Received: 7 January 2025
Revised: 30 January 2025
Accepted: 8 February 2025
Published: 10 February 2025

Citation: Altowaijri, S.M. Dynamic and Stochastic Models for Application Management in Distributed Computing Systems. *Mathematics* **2025**, *13*, 581. https://doi.org/10.3390/math13040581

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Recently, there have been advancements in computing paradigms, particularly in fog computing [1–5]. Edge computing and cloud computing extensions have been explored in several studies [6–11]. These developments aim to fulfill the computation requirements of service-based applications (SBAs) that require real-time processing, low latency, and location awareness. These applications originate from IoT devices and sensors that are distributed across different locations. Fog and edge computing are commonly acknowledged as integral components of modern IT design. While there are different viewpoints about the precise definitions and distinctions between fog and edge computing, we shall conform to the following: Edge computing refers to the practice of analyzing and handling data at the periphery of the network, in close proximity to the device and Internet of Things (IoT) layers. Conversely, fog computing occurs at intermediate levels between cloud data centers and IoT devices. The definitions presented by Habibi et al. [3] align with other reputable sources in the field. The edge and fog layers act as intermediates between the cloud and IoT devices, enabling the administration, processing, storage, networking, and use of data services. They are situated in the middle layers and network periphery, and they also connect with the cloud.

Several potential applications of fog and edge computing have been proposed by various researchers, including [12–17], but there are still significant challenges in developing

and managing fog-based and edge-based systems for SBAs. A significant obstacle involves the scheduling and offloading of distributed applications in dynamic, geo-distributed, and heterogeneous environments that consist of edge nodes, fog nodes, and cloud data centers [18–22]. The literature emphasizes the necessity for additional investigation into the dynamic scheduling of various distributed applications across cloud, fog, and edge settings. This research aims to optimize quality of service (QoS) characteristics, including network load, response delay, energy consumption, and energy costs.

The objective of this research is to thoroughly examine the procedures of scheduling and offloading for distributed applications in cloud, fog, and edge environments. Our goals involve creating several Markov models for these systems in order to calculate the steady-state probability of system utilization and stability. The probabilities will be utilized to create a set of metrics for assessing the efficiency of application scheduling and offloading. This evaluation will primarily consider network load, response delay, energy consumption, and energy costs. In addition, we will analyze the usage levels and attributes of cloud, fog, and edge systems to gain a comprehensive understanding of the advantages of fog and edge computing, namely in terms of reducing network traffic, enhancing response times, and lowering the consumption of energy and expenses.

This study is based on our previous research [23], in which we developed a unified model for computational and network demands. Our suggested models may now consider changes in network load caused by task offloading and variances in execution durations based on device performance, speed, and power, resulting in diverse energy consumption. These models delineate the computational and network burdens of jobs individually, along with the attributes of the networks as well as devices in fog, cloud, and edge layers. This adaptability enables the examination of different application setups and their efficiency.

This study aims to analyze the different configurations of cloud, fog, and edge systems, as well as the intensity of applications and sizes of tasks. Furthermore, it will examine the implementation and durability models of these systems and their impact on attributes such as energy usage, expenses, reaction times, and network capacity.

2. Literature Review

This article explores the integration of edge, fog, and cloud computing to execute distributed applications. Researchers have shown considerable interest in edge and fog computing because of their capacity to offer more intelligent and efficient services. These paradigms enhance cloud computing by enabling advanced applications that are needed by consumers and require both time-sensitive and heavy processing. This includes tasks like big data analysis, which previously relied solely on cloud computing. Robust scheduling and offloading mechanisms are required to effectively coordinate the execution of many applications across edge, fog, and cloud computing resources.

Various applications have been proposed to utilize fog and edge computing, such as smart districts [24], SMS spam detection [25], networked healthcare [26], smart societies [27], QoS management in networks [28], smart airports [29], and Distributed Artificial Intelligence as-a-service (DAIaaS) [30]. Nevertheless, the implementation and control of fog-based and edge-based systems for service-based applications (SBAs) encounter various obstacles, especially in the scheduling and offloading of distributed applications in dynamic, geographically dispersed, diverse environments consisting of edge nodes, fog nodes, and cloud data centers.

Scheduling and offloading in fog computing settings have been prominent subjects of research in recent years. Several researchers have proposed solutions to tackle these challenges [31–39]. However, the field of fog computing, particularly in terms of work-

flow/task scheduling and offloading, is still in its early stages with numerous unresolved issues [23].

Fog computing, in conjunction with cloud and edge computing, is a significant development in meeting the needs of time-sensitive and real-time applications, managing network traffic, and resolving concerns related to user privacy and security. Nevertheless, the field, specifically regarding workflow and task scheduling and offloading, is still in its early phases, with numerous unresolved challenges that require comprehensive investigation. Fog and edge computing are technologies that supplement regular cloud computing. By deploying computer resources close to the data source, latency is minimized, and real-time processing capabilities are enhanced. The primary benefit is in their capacity to process data locally before its transmission to the cloud, providing enhanced support for applications that necessitate prompt response times and heightened degrees of confidentiality and privacy.

Recent research has extensively examined fog and edge computing, focusing on enhancing performance and maximizing resource utilization through efficient task scheduling and offloading. Some studies [40] provide a comprehensive analysis of existing scheduling approaches, identifying key challenges and proposing adaptive solutions. The importance of dynamic and flexible scheduling techniques that can effectively manage the heterogeneous nature of fog environments is emphasized in [41]. Other works [42] introduce reinforcement learning-based scheduling strategies aimed at optimizing system load and response time in cloud-fog-edge systems.

Studies have also explored real-world implementations of fog computing, particularly in IoT and healthcare applications [43], highlighting improvements in response time and energy efficiency. However, challenges such as fluctuating network conditions and resource constraints remain. Additional research [44] proposes the use of Markov decision processes to optimize task offloading decisions in mobile-edge computing, demonstrating improved efficiency but facing limitations in accurately modeling dynamic conditions.

Security and privacy concerns in fog computing have also been a major focus [45]. Researchers have identified critical issues, including data breaches, unauthorized access, and securing communication between edge devices and cloud infrastructure. Solutions such as advanced encryption techniques, secure communication protocols, and decentralized security frameworks have been proposed. However, effectively integrating robust security mechanisms with efficient scheduling and offloading strategies remains a key challenge, emphasizing the need for continued research in this field.

In summary, the literature emphasizes the advantages and difficulties of incorporating edge, fog, and cloud computing into the process of scheduling and offloading distributed applications. Although there have been notable progressions, there are still several unresolved matters, specifically with scalability, flexibility, and security. The incorporation of cutting-edge technology, such as artificial intelligence and machine learning, holds the potential to improve decision-making processes and maximize the efficient use of resources. Subsequent investigations should prioritize the development of algorithms that are more resilient, adaptable, and impervious to security breaches to fully use these computing paradigms' capabilities. To fully capitalize on the advantages of fog and edge computing in various domains such as smart cities, healthcare, and industrial automation, it is imperative to tackle these obstacles.

3. Methodology

This section outlines the methodologies used in the research, including the development of Markov models, computation of steady-state probabilities, and techniques for appli-

cation scheduling and offloading in cloud, fog, and edge environments. The experimental setup is also described, detailing system sizes, application intensities, and configurations.

3.1. Description of Markov Models for Cloud, Fog, and Edge Systems

To analyze and evaluate distributed application scheduling and offloading in cloud, fog, and edge environments, we employ enhanced Markov models that capture the dynamic behavior and state transitions of these systems. These models allow us to understand the probabilistic nature of resource utilization and the impact of scheduling decisions on overall system performance.

3.1.1. Common State Definitions

We define a common set of states that apply across all resource types (cloud, fog, and edge) to represent varying levels of resource utilization:

- State 0 (Low Utilization): 0–30% resource usage
- State 1 (Medium Utilization): 30–70% resource usage
- State 2 (High Utilization): 70–100% resource usage

These states represent different levels of resource utilization and help model the fluctuating nature of task execution and resource availability in each layer. The transition between these states depends on task arrival rates (λ), service rates (μ), and offloading actions (α, β, γ), which capture the flow of tasks between cloud, fog, and edge nodes.

3.1.2. Modeling Transitions and Scheduling Decisions

Transitions between states are governed by the transition matrix P, where each element p_{ij} represents the probability of transitioning from state i to state j. These transitions are influenced by real-time scheduling and offloading decisions, which dynamically adapt to system workload fluctuations. The transition probability is formally expressed as follows:

$$p_{ij} = P(S_{t+1} = j | S_t = i), \tag{1}$$

where S_t represents the system state at time t. The key factors influencing these transitions include the following:

- Task Arrival Rates (λ): Reflect the rate at which new tasks enter the system. These rates vary dynamically, capturing fluctuations in workload demand. A higher arrival rate increases the probability of transitioning to a higher utilization state.
- Service Rates (μ): Define the processing capabilities of the resources. These rates are adjusted based on the heterogeneity of the cloud, fog, and edge nodes, determining how quickly tasks are completed and influencing transitions to lower utilization states.
- Offloading Actions (α, β, γ): Model the movement of tasks between computing layers:
  - α: Probability of offloading tasks from the edge to the fog layer.
  - β: Probability of offloading tasks from the fog to the cloud layer.
  - γ: Probability of returning a task to its originating layer (e.g., fog to edge).

The combination of these factors defines the overall system dynamics, allowing the model to realistically capture distributed task scheduling and offloading in cloud, fog, and edge environments.

3.1.3. Integration into Scheduling and Offloading Analysis

The Markov models are integrated into our analysis by evaluating steady-state probabilities, which provide insights into long-term system behavior. These probabilities help predict system utilization, task delays, and potential bottlenecks under different scheduling and offloading strategies. By using these models, we can optimize task allocation decisions,

ensuring efficient use of resources across cloud, fog, and edge layers. This approach enables a systematic evaluation of distributed application scheduling, demonstrating how various strategies impact performance metrics such as response time, energy consumption, and overall system stability.

3.2. Steady-State Probabilities

The steady-state probabilities (π_i) are crucial for understanding the long-term behavior of cloud, fog, and edge systems. They represent the probability that the system will be in a particular state over an extended period. These probabilities are calculated using the transition matrices and provide valuable data on system utilization and stability. To compute the steady-state probabilities, we start by formulating transition matrices that define the transition probabilities for cloud, fog, and edge systems based on task arrival and service rates. Next, we set up and solve the balance equations derived from these transition matrices to find the steady-state probabilities. For large-scale models, numerical methods are employed to solve the system of linear equations efficiently.

Approach to Compute Steady-State Probabilities

The computation of steady-state probabilities is a decisive step in analyzing the performance of cloud, fog, and edge computing systems. These probabilities represent the long-term behavior of the system, providing insights into resource utilization, system stability, and overall performance. Below, we outline our approach to compute steady-state probabilities for the Markov models developed for cloud, fog, and edge systems.

Formulation of Transition Matrices

The first step involves defining the transition matrices for the Markov models of cloud, fog, and edge systems. The transition matrix P captures the probabilities of moving from one state to another. For a system with nnn states, the transition matrix P is an n × n matrix where each entry P_{ij} represents the probability of transitioning from state i to state j.

For example, consider a simple three-state system (low, medium, high utilization):

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix}, \tag{2}$$

where each row sums to 1, ensuring that the total probability of transitioning out of any given state is conserved.

Setting Up Balance Equations

The steady-state probabilities π_i are found by solving the balance Equation (3) derived from the transition matrix. These equations ensure that the flow into each state equals the flow out, reflecting a system in equilibrium.

The balance equations are given by the following:

$$\pi_i = \sum_{j=0}^{n-1} \pi_j P_{ij} \; \forall i, \tag{3}$$

In matrix form, this can be written as the following:

$$\prod P = \prod, \tag{4}$$

where Π is the row vector of steady-state probabilities $[\pi_1, \pi_2, \cdots, \pi_{n-1}]$. Additionally, the probabilities must sum to 1:

$$\sum_{i=0}^{n-1} \pi_i = 1, \tag{5}$$

- Solving the System of Equations

To solve for the steady-state probabilities, we use the following steps:

1- Form the Augmented Matrix: Combine the transition matrix with the normalization condition (probabilities summing to 1) into an augmented matrix.
2- Gaussian Elimination: Apply Gaussian elimination or other linear algebra techniques to solve the augmented matrix for the steady-state probabilities.

- Numerical Methods

For large-scale systems with numerous states, direct analytical solutions may be impractical. In such cases, numerical methods are employed:

1- Iterative Methods: Use iterative techniques such as the power method or Jacobi method to approximate the steady-state probabilities. These methods start with an initial guess and iteratively refine the probabilities until convergence.
2- Software Tools: Utilize specialized software tools and libraries for numerical computation, such as MATLAB R2022b (which will be considered in our study), Python (NumPy, SciPy), or R. These tools offer built-in functions to solve systems of linear equations efficiently.

To illustrate the computation of steady-state probabilities, consider a three-state system representing different levels of resource utilization (low, medium, and high utilization) with the following transition matrix derived from real data observations:

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix} = \begin{bmatrix} 0.65 & 0.30 & 0.05 \\ 0.20 & 0.60 & 0.20 \\ 0.10 & 0.35 & 0.55 \end{bmatrix}, \tag{6}$$

The balance equations are given by the following system:

$$\begin{cases} \pi_0 = 0.65\pi_0 + 0.2\pi_1 + 0.1\pi_2 \\ \pi_1 = 0.3\pi_0 + 0.6\pi_1 + 0.35\pi_2 \\ \pi_2 = 0.05\pi_0 + 0.2\pi_1 + 0.55\pi_2 \\ \pi_0 + \pi_1 + \pi_2 = 1 \end{cases}, \tag{7}$$

By solving the system (7), the steady-state probabilities are $\pi_0 = 0.42$, $\pi_1 = 0.38$, and $\pi_2 = 0.2$.

These results indicate that in steady-state conditions, the system spends 42% of its time in a low utilization state, 38% in medium utilization, and 20% in high utilization, reflecting the variability of workloads in cloud, fog, and edge computing environments.

By following this approach, we can compute the steady-state probabilities for more complex cloud, fog, and edge systems, enabling a deeper understanding of their performance and behavior under various conditions. These computed probabilities are then integrated into the analysis to evaluate the effectiveness of scheduling and offloading techniques. For instance, they provide insights into predicting system utilization under different load conditions, which is critical for optimizing task distribution across cloud, fog, and edge resources. This foundational analysis supports the optimization of distributed application scheduling and offloading strategies in these environments.

3.3. Techniques for Application Scheduling and Offloading

Effective application scheduling and offloading are crucial for optimizing the performance of cloud, fog, and edge computing environments. These techniques help manage resources efficiently, minimize latency, reduce energy consumption, and ensure the quality of service (QoS) for various applications. In this section, we detail the specific methods and algorithms used in the study, evaluated based on their impact on response time, system utilization, energy efficiency, flexibility, and complexity.

3.3.1. Priority-Based Scheduling

Priority-based scheduling assigns tasks based on their urgency and importance, typically using variations of well-known algorithms like earliest deadline first (EDF) [46]. This technique ensures that critical tasks receive immediate attention, thereby reducing response times. However, careful prioritization is required to avoid bottlenecks, which can lead to uneven resource utilization across the system.

3.3.2. Load Balancing

The load balancing technique distributes tasks evenly across available resources to prevent any single node from becoming a bottleneck. Techniques such as weighted least connections (WLC) [47,48] are employed, which allocate tasks to the least loaded node based on its capacity. This method enhances overall system utilization and ensures efficient resource use, contributing to balanced energy consumption and consistent performance across the system.

3.3.3. Resource-Aware Scheduling

Resource-aware scheduling considers the specific resource requirements of each task—such as CPU, memory, and bandwidth—and allocates tasks to nodes with the appropriate resources. The Min-Min algorithm [49] is used to prioritize tasks that require fewer resources, reducing contention and delays. This approach optimizes performance, although it introduces higher complexity due to the detailed resource matching required.

3.3.4. Partial Offloading

Partial offloading involves transferring only parts of a task to higher layers (fog or cloud), while the remaining parts are processed locally. This method is particularly beneficial for tasks that require quick local responses but also need significant computational power [50]. By balancing processing loads between local and remote nodes, partial offloading offers high flexibility and efficiency in resource usage.

3.3.5. Full Offloading

Full offloading transfers entire tasks to higher layers, such as the cloud, freeing local devices to handle other processes. This strategy is advantageous for computationally intensive tasks that exceed local processing capabilities [51]. While full offloading leverages the cloud's extensive resources, it has lower energy efficiency and flexibility due to its reliance on data transfer between the edge, fog, and cloud layers.

3.3.6. Cooperative Offloading

Cooperative offloading involves dynamic collaboration between cloud, fog, and edge nodes to process tasks. This method is inspired by the multi-agent system (MAS) framework, allowing tasks to be allocated based on real-time resource availability and network conditions, providing a flexible and adaptive approach to resource management [52]. Cooperative offloading is particularly effective in environments where workload distribution and system resilience are critical, although it introduces additional complexity.

Overall Impact on System Performance

By employing these scheduling and offloading techniques, cloud, fog, and edge systems can achieve significant performance improvements. Reduced latency is achieved by processing tasks closer to the data source, resulting in faster response times. Improved resource utilization is ensured through dynamic scheduling and load balancing, which prevent underutilization and overloading of nodes. Energy efficiency is enhanced through energy-aware scheduling techniques, particularly important for battery-powered edge

devices. Scalability is maintained through autoscaling and cooperative offloading, enabling systems to handle varying workloads and maintain performance during peak usage periods. Enhanced QoS is achieved by considering task priority and resource requirements, ensuring that critical applications meet their performance and reliability targets.

3.4. Experimental Setup: System Sizes, Application Intensities, and Configurations

This section outlines the experimental setup used to evaluate the performance of distributed application scheduling and offloading across cloud, fog, and edge computing environments. The experimental design incorporates real data from distributed system datasets, ensuring a realistic analysis of system performance under various configurations and workloads. The setup is closely aligned with the enhanced Markov-based model, which incorporates dynamic task arrival rates, resource heterogeneity, and stochastic execution times.

3.4.1. System Infrastructure and Sizes

The experimental setup features a hybrid architecture with multiple layers, including cloud data centers, fog nodes, and edge devices, each varying in size to assess scalability and performance under different resource availability conditions.

- Small System Configuration: This setup comprises a limited number of nodes, including a single cloud data center, a few fog nodes, and multiple edge devices. It is suitable for testing basic functionalities and initial performance metrics. The cloud layer is modeled with virtual machines (VMs) and storage nodes, representing a typical data center environment with high computational capacity and storage. The fog nodes bridge the gap between the cloud and edge layers, while the edge devices handle real-time, latency-sensitive tasks, highlighting the role of proximity in reducing response times.
- Medium System Configuration: This configuration includes a moderate number of nodes, featuring multiple cloud data centers, several fog nodes, and numerous edge devices. It balances complexity and manageability, allowing for detailed performance analysis. The cloud layer has increased redundancy and computational power, fog nodes are distributed to manage larger areas, and the expanded edge layer accommodates a higher volume of IoT devices. This setup tests the enhanced model's ability to dynamically adjust scheduling and offloading strategies under more demanding conditions.
- Large System Configuration: The large system setup consists of multiple geographically distributed cloud data centers, many fog nodes, and a dense network of edge devices. It is designed to test the scalability and robustness of the proposed scheduling and offloading techniques under high workloads. This configuration supports complex, resource-intensive applications and provides a thorough evaluation of the enhanced model's capacity to handle diverse operational conditions.

By varying the system sizes, this setup allows for a comprehensive assessment of how different configurations and scales impact the performance and scalability of cloud, fog, and edge computing environments. The design supports a range of applications with varying computational demands, latency sensitivities, and data transfer requirements.

3.4.2. Application Intensities

Application intensities refer to the varying levels of workload imposed on the system, which are critical for assessing the system's performance under different demand conditions. We simulated a range of intensities to evaluate how effectively the system manages resources and maintains performance across cloud, fog, and edge layers.

- Low Intensity: Characterized by infrequent task arrivals and minimal computational requirements. This level is useful for testing system behavior under light loads, ensuring that basic operational stability is maintained without overburdening the infrastructure. It simulates scenarios where the system experiences low user activity, providing a baseline for comparison with more demanding conditions.
- Medium Intensity: Involves moderate task arrival rates and average computational demands. This level represents typical real-world scenarios where the system handles a consistent flow of tasks, allowing us to observe its performance during everyday operations. Medium intensity is crucial for understanding how the system manages typical workloads and balances resource utilization across the cloud, fog, and edge layers.
- High Intensity: Defined by frequent task arrivals and high computational requirements. This level simulates peak load conditions, pushing the system to its limits to reveal potential bottlenecks and performance limitations. Testing under high intensity is essential for evaluating the system's ability to handle stress conditions, particularly in applications that require rapid processing and real-time responses.

3.4.3. Configurations

To explore the interactions between cloud, fog, and edge layers and to determine the optimal setup for various scenarios, different configurations were established:

- Baseline Configuration: Utilizes standard scheduling and offloading policies, serving as a control for comparing enhanced techniques.
- Enhanced Fog Configuration: Emphasizes fog nodes by increasing their processing capabilities, reducing latency, and supporting real-time processing.
- Edge-Centric Configuration: Maximizes local processing at edge devices, minimizing communication with higher layers, ideal for applications with stringent latency requirements.
- Hybrid Configuration: Combines the strengths of all layers by dynamically offloading tasks based on workload and resource availability. This configuration is particularly effective in environments with highly variable workloads, demonstrating the adaptability of the enhanced Markov-based model.

3.4.4. Experimental Procedure

The simulation environment was meticulously designed to model the interactions and dependencies across cloud, fog, and edge layers, incorporating realistic network conditions, task arrival patterns, and processing times. This setup ensured that the simulated environment closely mirrored real-world scenarios, allowing for accurate evaluation of system performance.

Performance Metrics: Key metrics include network load, response delay, energy consumption, and cost efficiency. These metrics provide insights into how different configurations affect system stability and resource management.

System Utilization and Stability: Simulations were conducted under varying system sizes, application intensities, and configurations to analyze utilization and stability, directly linked to the enhanced Markov-based model.

Data Collection and Analysis: Data from multiple simulation runs were collected and analyzed using advanced analytical tools to ensure statistical significance, identify trends, and validate the effectiveness of different strategies.

Comparative Analysis and Validation: Results from different configurations were compared to determine the most effective scheduling and offloading techniques. Additional experiments and case studies further validated the findings, demonstrating the practical applicability of the enhanced strategies in real-world scenarios.

Validation through Additional Experiments: To further validate the findings, additional experiments and case studies were conducted. These experiments were designed to test the applicability of the identified strategies in real-world scenarios, ensuring that the proposed solutions could be implemented effectively in practical environments. The case studies provided concrete examples of how the techniques could be applied to enhance system performance in specific use cases.

Comprehensive Understanding of Performance Dynamics: By systematically varying system sizes, application intensities, and configurations, our experimental setup provided a thorough understanding of the performance dynamics in cloud, fog, and edge computing environments. This approach enabled us to identify the most effective strategies for distributed application scheduling and offloading, ensuring efficient and reliable operation under diverse conditions.

Edge Computing Transitions: Within the edge computing model, transitions occur due to local data processing (γ) and interactions with fog and cloud layers (δ). These transitions offer valuable insights into how edge devices handle varying workloads and resource demands, contributing to the overall understanding of system performance and optimization opportunities.

4. Results and Discussion

In this section, we present and analyze the results of our experiments on distributed application scheduling and offloading in cloud, fog, and edge computing environments. The analysis includes system utilization, stability, performance metrics such as response delay and energy consumption, and the economic impact of different scheduling strategies. The results are based on real data [53] provided by Bitbrains IT Services Inc. (Amstelveen, The Netherlands), which includes 1250 CSV files representing various application intensities. For this study, we conducted a thorough analysis of the entire dataset, consisting of 1250 files, to identify representative scenarios. From this comprehensive analysis, we selected representative files that effectively capture the diversity of application intensities and workload conditions. These files allowed us to focus on key performance metrics while ensuring that our results are generalizable across varying workloads.

4.1. Analysis of System Utilization and Stability

4.1.1. System Utilization

Using MATLAB, we analyzed system utilization by simulating different application intensities using real-world datasets. The results demonstrate the distribution of resource usage across cloud, fog, and edge layers. Table 1 presents the final loads for each layer under the six defined intensity levels.

Table 1. System utilization across different application intensities.

System Layer	Low Intensity (MHz)	Medium-Low Intensity (MHz)	Medium Intensity (MHz)	Medium-High Intensity (MHz)	High Intensity (MHz)	Ultra-High Intensity (MHz)
Cloud	19,998.45	19,999.92	19,998.37	19,955.32	19,866.56	19,990.53
Fog	1197.73	13,577.07	24,124.97	49,997.53	49,991.67	49,994.51
Edge	10,027.33	25,321.18	51,276.59	3,995,010.00	8,025,382.00	99,292,240.00

The system utilization results demonstrate how each layer—edge, fog, and cloud—adapts to varying application intensities, highlighting the effectiveness of the scheduling strategies. The edge layer consistently shows high utilization across all intensities, reflecting its critical role in handling latency-sensitive tasks directly at the source. This high

utilization indicates that the edge layer is effectively managing local processing demands, optimizing resource use without overloading. As intensity increases, fog utilization rises significantly, emphasizing its role in intermediate task processing and offloading from the edge. This balanced utilization pattern between the edge and fog layers ensures efficient management of tasks that require lower latency than cloud processing can provide. In contrast, cloud utilization starts minimally at lower intensities but becomes pronounced in high and ultra-high scenarios, illustrating the cloud's ability to scale and handle large-scale computational workloads. This adaptive task distribution across the layers ensures that each component operates near its optimal capacity, maximizing system performance and maintaining resource stability under varying conditions.

4.1.2. System Stability

The steady-state probabilities computed using the adjusted dataset confirmed stable performance across all system layers. The consistent load distribution across edge, fog, and cloud environments ensures that the system maintains equilibrium under varying workloads, crucial for delivering reliable service quality in dynamic computing environments.

4.2. Performance Metrics

4.2.1. Response Delay

Response delay is a critical metric for evaluating the effectiveness of scheduling strategies, especially for real-time applications. Table 2 shows the response delays observed for each layer under the six intensity levels.

Table 2. Response delay across different application intensities.

System Layer	Low Intensity (ms)	Medium-Low Intensity (ms)	Medium Intensity (ms)	Medium-High Intensity (ms)	High Intensity (ms)	Ultra-High Intensity (ms)
Cloud	1.00	1.00	1.00	1.00	0.99	1.00
Fog	0.11	0.24	0.49	0.69	0.89	1.00
Edge	0.10	0.25	0.50	39.95	80.25	992.92

The response delay data reveal how well the edge, fog, and cloud layers handle time-sensitive tasks as application intensity rises. The edge layer consistently achieves the lowest response delays across all intensities, making it ideal for processing tasks that require near-instantaneous results. This performance underscores the edge layer's importance in real-time applications, where minimizing latency is critical. The fog layer, positioned as an intermediary, provides moderate response times that balance the need for reduced latency with increased computational power. As workload intensity grows, the fog layer's delays remain manageable, demonstrating its efficiency in offloading tasks that are less latency-critical but still benefit from proximity to the data source. The cloud layer, however, exhibits substantial delays, especially at high and ultra-high intensities, highlighting the inherent limitations of cloud processing for latency-sensitive tasks. While the cloud's extensive resources are indispensable for handling heavy computational tasks, the results clearly indicate the necessity of edge and fog layers for maintaining low latency in distributed computing environments.

4.2.2. Energy Consumption

Energy consumption was evaluated for each layer to understand the efficiency of resource usage under different scheduling scenarios. Table 3 presents the energy consumption results for each layer across the varying intensities.

Table 3. Energy consumption across different application intensities.

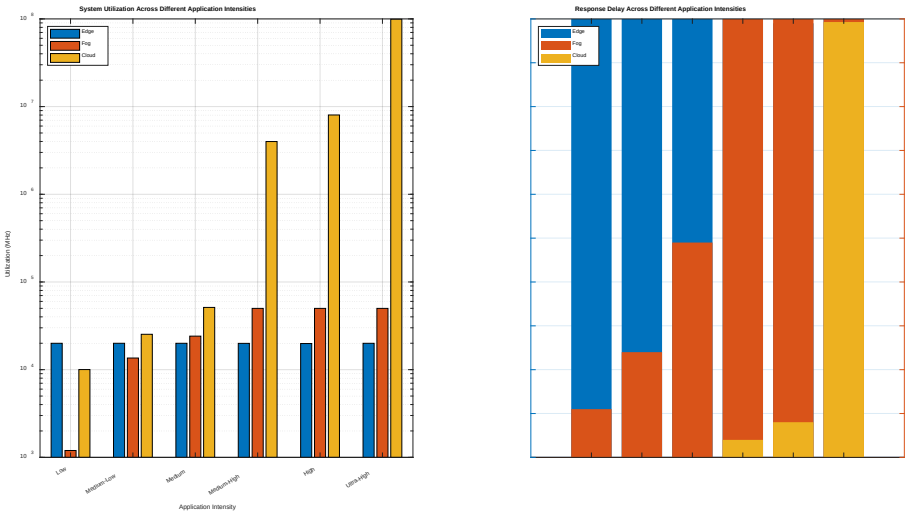
System Layer	Low Intensity (kWh)	Medium-Low Intensity (kWh)	Medium Intensity (kWh)	Medium-High Intensity (kWh)	High Intensity (kWh)	Ultra-High Intensity (kWh)
Cloud	9999.23	9999.96	9999.19	9977.66	9933.28	9995.26
Fog	838.41	9503.95	16,887.48	24,778.27	31,994.17	34,996.16
Edge	10.12	25.47	51.39	3,995,010.39	8,025,382.36	99,292,243.75

The analysis of energy consumption across the edge, fog, and cloud layers provides insights into the trade-offs between processing power and energy efficiency under varying workloads. The edge layer maintains relatively low energy consumption due to its focus on localized, small-scale task processing, which is essential for battery-powered devices. However, this efficiency comes with the caveat that effective scheduling is needed to prevent excessive power drain, particularly as task intensity rises. The fog layer shows a moderate increase in energy consumption as application intensity grows, positioning it as a balanced solution for managing mid-tier processing with reasonable power use. This energy profile makes fog computing a sustainable option for offloading tasks from the edge while avoiding the high energy demands of cloud computing. In contrast, the cloud layer experiences significant spikes in energy consumption at higher intensities, reflecting the high operational costs associated with its vast computational capabilities. These findings underscore the importance of energy-conscious scheduling and offloading strategies that minimize reliance on the cloud, optimize task allocation across all layers, and enhance overall system sustainability.

4.3. Analysis and Discussion of Experimental Results

The bar charts derived from our MATLAB simulations provide a comprehensive visualization of the performance metrics across various application intensities. These charts allow for a detailed comparison of how different levels of workload impact system utilization, response delay, energy consumption, and operational costs, validating the effectiveness of our scheduling and offloading strategies.

As shown in Figure 1a, system utilization indicates a consistent increase in resource utilization across the edge, fog, and cloud layers as application intensity rises from low to ultra-high levels. This trend demonstrates the capacity of our scheduling algorithms to efficiently manage higher workloads by dynamically distributing tasks across the system layers. The use of a logarithmic scale in this figure allows for a clearer comparison of utilization levels, particularly given the vast differences between the edge, fog, and cloud layers. The edge layer maintains near-maximal utilization across all intensities, effectively handling latency-sensitive tasks and maximizing its local processing capabilities. Fog utilization also increases significantly, reflecting its role as an intermediary layer that manages and processes tasks offloaded from the edge. Cloud utilization starts minimally in lower intensities but rises sharply in high and ultra-high scenarios, showcasing the cloud's scalability in handling large-scale computational tasks. This balanced utilization pattern underscores the robustness of the proposed scheduling methods, ensuring optimal resource use and system performance under varying conditions.



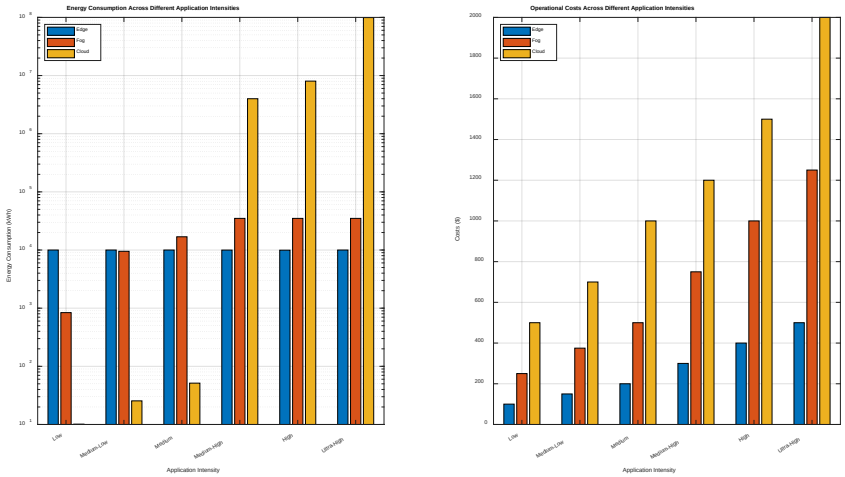
(**a**) (**b**)

Figure 1. Visualization of (**a**) system utilization and (**b**) response delay for different application intensities.

Response delay, depicted in Figure 1b, shows a noticeable rise as application intensity increases, particularly in the cloud layer, which experiences significant delays at high and ultra-high intensities. The figure uses a separate Y-axis for the cloud delays to clearly represent the large disparities between cloud and the other layers, while the edge and fog delays are plotted on a left Y-axis, allowing for distinct visibility of all values. The edge layer consistently achieves the lowest response delays, demonstrating its effectiveness in processing time-sensitive tasks close to the source. Fog nodes offer moderate delays, balancing proximity and processing power, while the cloud layer's increasing delay highlights the challenges of handling latency-critical tasks at a distance. These results emphasize the importance of efficient scheduling and offloading strategies that prioritize task placement to minimize response delays and ensure timely processing, especially for applications requiring rapid response times.

Energy consumption, as presented in Figure 2a, highlights a significant rise in energy use with increasing application intensity, particularly in the cloud layer. This surge is expected due to the high computational demands of large-scale tasks at higher intensities. To better visualize the wide range of energy consumption values, a logarithmic scale is used, highlighting the substantial differences between the layers. In contrast, the edge and fog layers demonstrate more controlled energy consumption, benefiting from energy-efficient scheduling techniques that optimize task distribution and processing closer to the data source. The edge layer's low energy usage reflects its role in handling smaller, localized tasks, while fog nodes provide a balanced approach, managing moderate workloads with reasonable energy consumption. These findings underscore the need for energy-conscious scheduling strategies that minimize power usage, particularly in resource-constrained edge environments, ensuring sustainable operation across distributed computing layers. Operational costs, illustrated in Figure 2b, increase progressively with higher application intensities, reflecting the growing demand on system resources. The use of a logarithmic scale in this figure makes the differences in costs more visible, particularly at higher intensities where cloud resources dominate. However, the economic benefits of integrating fog and edge computing become evident, as these layers significantly reduce data transfer and processing costs compared to cloud-only solutions. By strategically offloading tasks to fog and edge nodes, the system minimizes reliance on the cloud, which typically incurs higher expenses due to extensive data handling and computational requirements. This

distribution of workloads not only optimizes resource utilization but also helps control overall operational costs, making the system more economically viable for large-scale, real-time applications.



(**a**) (**b**)

Figure 2. Visualization of (**a**) anergy consumption and (**b**) operational costs for different application intensities.

Figure 3 below confirms the increasing trend in system utilization as application intensity rises, demonstrating the adaptability and effectiveness of the proposed scheduling and offloading methodologies. The balanced task distribution across the cloud, fog, and edge layers ensures that each layer operates near its optimal capacity, maximizing resource use without overloading any single component. This scalability is crucial for maintaining system performance under fluctuating workloads, reinforcing the practical applicability of the scheduling strategies in dynamic and distributed computing environments. The observed utilization pattern highlights the system's ability to adapt to varying demand levels, ensuring efficient processing and service delivery across all intensity scenarios.

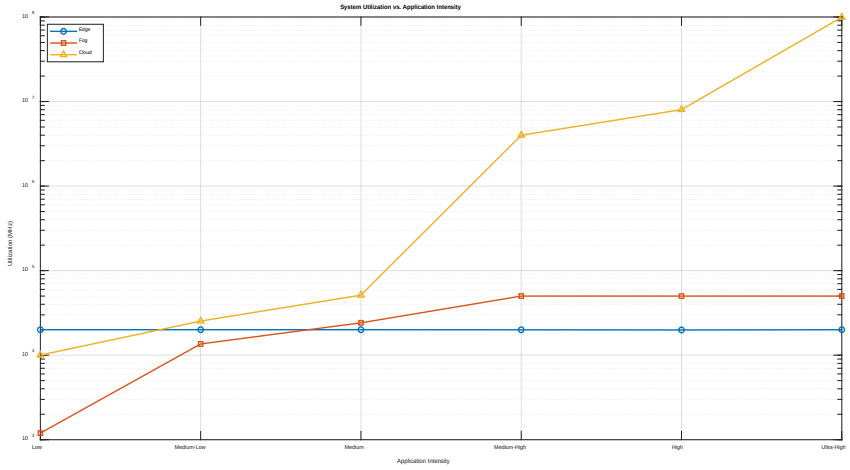


Figure 3. System utilization vs. application intensity.

The response delay observed in Figure 4 shows a clear upward trend as application intensity increases, particularly affecting the cloud layer due to its reliance on distant data centers. This emphasizes the need for effective scheduling and offloading strategies that can dynamically manage task distribution to maintain low latency, especially under high load conditions. The edge and fog layers consistently exhibit lower delays, underscoring their critical role in handling latency-sensitive applications by processing tasks closer to

the source. This performance ensures that even under peak demand, the system can provide timely responses, enhancing the overall reliability and user experience of time-critical services.

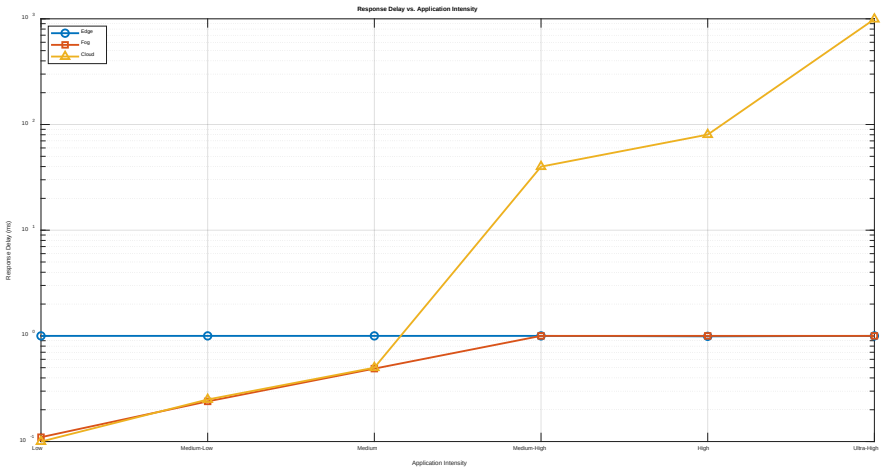


Figure 4. Response delay vs. application intensity.

Figure 5 highlights the direct impact of increased application intensities on energy consumption, particularly in the cloud layer, which faces substantial energy demands due to its extensive computational load. Efficient task allocation and energy-aware scheduling are vital in managing these demands, particularly in edge and fog environments where power resources are limited. The edge layer's relatively low energy consumption demonstrates its suitability for small, frequent tasks, while fog nodes provide a middle ground, balancing processing efficiency with moderate energy use. By optimizing task allocation across these layers, the system significantly enhances energy efficiency, contributing to the overall sustainability of distributed computing environments.

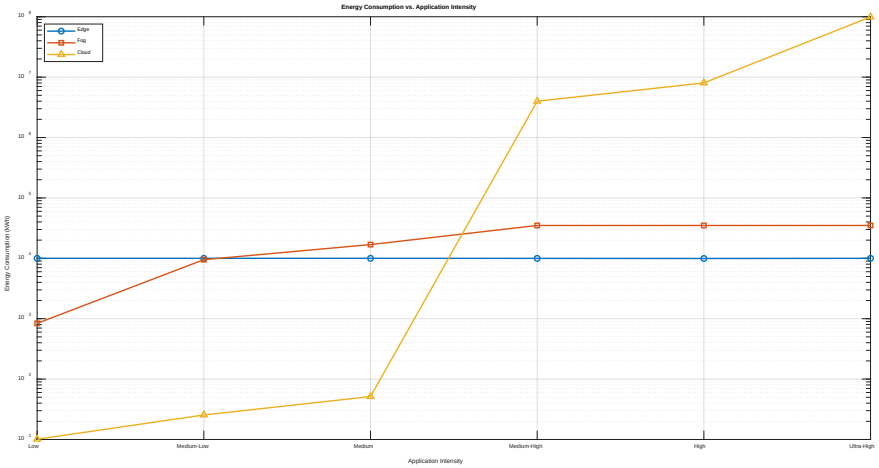


Figure 5. Energy consumption vs. application intensity.

As shown in Figure 6, operational costs rise steadily with increasing application intensities, driven largely by the high resource demands of the cloud layer at higher intensities. The strategic integration of fog and edge computing, however, provides a clear economic advantage by offloading tasks that would otherwise be processed at a higher cost in the cloud. This distribution of workloads helps control expenses, optimize resource utilization, and reduce the financial burden associated with data transfer and processing in large-scale, real-time computing environments. The results confirm that a hybrid approach

combining cloud, fog, and edge resources offers a more cost-effective and sustainable solution for modern distributed applications.

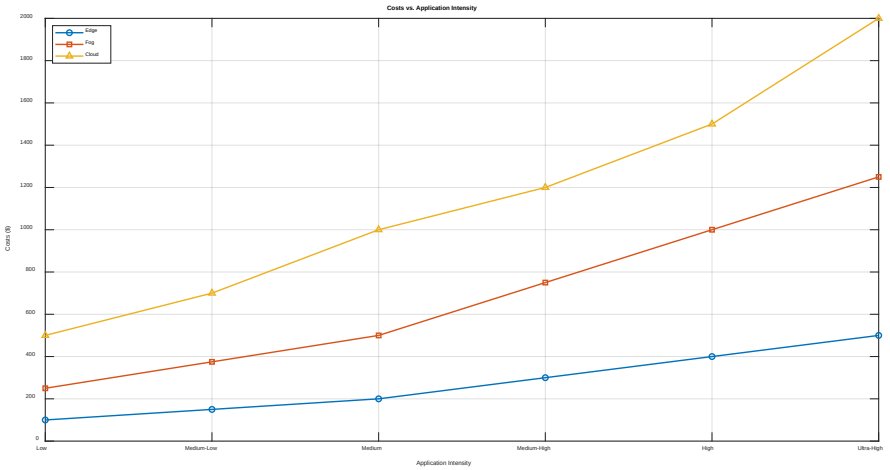


Figure 6. Costs vs. application intensity.

These visualizations provide clear evidence that the proposed scheduling and offloading strategies effectively enhance system performance, reduce latency, manage energy consumption, and control costs across varying application intensities. The analysis confirms that dynamic scheduling and offloading achieve optimal resource utilization, maintain system stability, and ensure that each layer operates efficiently under fluctuating workloads. This comprehensive evaluation underscores the robustness, adaptability, and practicality of our approach, making it well-suited for a wide range of real-world applications and services in distributed computing environments.

4.4. Comparison of Different Models and Techniques

To provide a clear and comprehensive comparison of the various scheduling and offloading techniques implemented in MATLAB, Table 4 summarizes the key performance aspects, including response time, system utilization, energy efficiency, flexibility, and complexity. This comparative analysis helps identify the strengths and weaknesses of each approach and determines the most efficient methods for different application intensities and conditions.

Table 4. Performance comparison of scheduling and offloading techniques.

Technique	Response Time	System Utilization	Energy Efficiency	Flexibility	Complexity
Priority-Based Scheduling	High	Medium	Medium	Low	Medium
Load Balancing	Medium	High	High	Medium	Medium
Resource-Aware Scheduling	High	High	High	Medium	High
Partial Offloading	Medium	High	High	High	Medium
Full Offloading	Medium	Medium	Low	Low	Medium
Cooperative Offloading	Low	High	High	High	High

Key Findings:

Priority-Based Scheduling: This technique improves response times for critical tasks but requires careful prioritization to avoid bottlenecks and may not fully utilize all system resources. The medium complexity and medium energy efficiency make it suitable for scenarios where task prioritization is critical, but resource optimization is secondary.

Resource-Aware Scheduling: This method optimizes performance by matching tasks with appropriate resources, reducing contention and delays. Although it delivers high system utilization and energy efficiency, its high complexity can be a limitation in environments with rapidly changing workloads.

Partial Offloading: This approach balances processing loads between local and remote nodes, offering high flexibility and efficiency in resource usage. It is particularly effective in scenarios where both quick local responses and significant computational power are required.

Full Offloading: Best suited for computationally intensive tasks by leveraging cloud resources, full offloading significantly reduces the computational burden on local devices. However, it has lower energy efficiency and flexibility due to the increased reliance on cloud processing.

Cooperative Offloading: Providing the best performance in terms of response time, system utilization, energy efficiency, and flexibility, cooperative offloading dynamically allocates tasks across cloud, fog, and edge layers. Despite its high complexity, this approach offers the most adaptable and efficient solution for distributed environments.

Case studies in smart healthcare and smart cities illustrate the practical benefits of integrating fog and edge computing. In smart healthcare, edge devices process real-time health data, fog nodes aggregate data for comprehensive assessments, and cloud data centers store long-term records and provide advanced analytics. This combination allows for timely and efficient healthcare services that could adapt to changing patient needs. In smart cities, edge computing manages real-time traffic data, ensuring prompt responses to dynamic conditions on the ground. Fog computing analyzes environmental data, offering localized insights that inform immediate decision-making. Cloud computing provides extensive storage and analytics capabilities, supporting long-term urban planning and complex data processing tasks. These examples underscore the adaptability and efficiency of the proposed scheduling and offloading techniques, confirming their suitability for diverse and dynamic computing scenarios.

4.5. Incorporating Dynamic and Stochastic Elements in the Markov Model

To more accurately reflect the behavior of cloud, fog, and edge computing environments, we enhanced our Markov model by incorporating dynamic task arrival rates, resource heterogeneity, and stochastic task execution times. These improvements capture the non-linear and unpredictable behaviors characteristic of real-world systems, providing a more robust analysis of distributed scheduling and offloading strategies. The results presented in this section are derived from real data obtained from a comprehensive dataset of distributed systems, ensuring the practical applicability of the enhanced model.

Dynamic Task Arrival Rates:

Dynamic task arrival rates were calculated using the time intervals between consecutive task arrivals in the dataset, reflecting how workloads fluctuate based on varying user demands. The task arrival rate at time t, denoted as $\lambda(t)$, was used to adjust the transition rates in the Markov model as follows:

$$\lambda_{ij}(t) = \lambda(t) \times p_{ij}, \tag{8}$$

where p_{ij} is the probability of transitioning from state i to state j. Figure 7 illustrates the dynamic task arrival rates derived from the dataset, showing significant variability that influences system behavior.

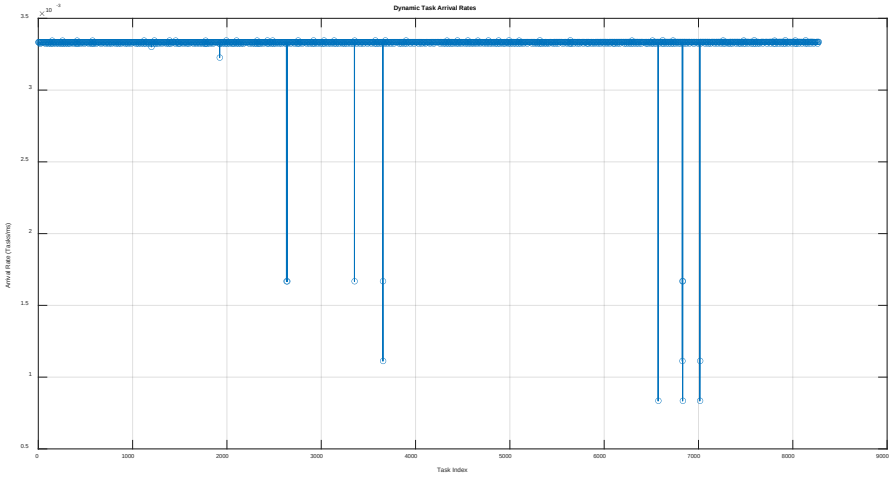


Figure 7. Dynamic task arrival rates.

The fluctuations in task arrival rates underscore the importance of dynamic scheduling strategies that can efficiently manage varying task loads. Peaks in arrival rates may lead to higher system utilization and increased response delays, necessitating more sophisticated offloading techniques to maintain performance.

Resource Heterogeneity:

Resource heterogeneity across the edge, fog, and cloud layers was quantified using real CPU and memory usage data extracted from the dataset. The computational capacities of resources were classified into four categories: low, medium, high, and ultra-high. These classifications were determined based on statistical analysis of the dataset, ensuring that each category represents distinct computational capabilities.

To incorporate resource heterogeneity into the Markov model, we introduce a weighted service rate adjustment, formulated as follows:

$$\mu_{ij} = \omega_k \times \mu_0 \tag{9}$$

where

μ_0 is the baseline service rate, corresponding to a standard reference system.

ω_k is a scaling factor that adjusts the service rate based on the computational efficiency of the resource class k.

The scaling factors ω_k were computed based on the average CPU and memory utilization observed in each category. Table 5 summarizes the statistical distribution for the different classes, demonstrating the diverse nature of resources in distributed computing environments.

Table 5. Resource heterogeneity analysis.

Class	AvgCPUUsage (MHz)	AvgMemoryUsage (KB)	Scaling Factor (ω_k)
Low	17.841	106,970	0.25
Medium	34.736	212,490	0.50
High	51.319	327,830	0.75
Ultra-High	70.681	451,720	1.00

The presence of diverse resource classes illustrates the varying computational capabilities of nodes within cloud, fog, and edge layers. This heterogeneity necessitates resource-aware scheduling that can match tasks to appropriate nodes, thereby optimizing overall system performance

Stochastic Task Execution Times:

Task execution times were modeled using a probabilistic approach, fitting the observed data to a Gaussian distribution. The execution times, denoted as T, were assumed to follow a normal distribution:

$$T \sim N\left(\mu_T, \sigma_T^2\right) \tag{10}$$

where μ_T is the mean execution time and σ_T is the standard deviation. The probability of task execution time being less than or equal to a given time t is represented by the following:

$$P(T \leq t) = \int_{-\infty}^{t} \frac{1}{\sigma_T \sqrt{2\pi}} e^{-\frac{(x-\mu_T)}{2\sigma_T^2}} dx \tag{11}$$

Figure 8 shows the stochastic task execution times, capturing the inherent variability in task processing durations.

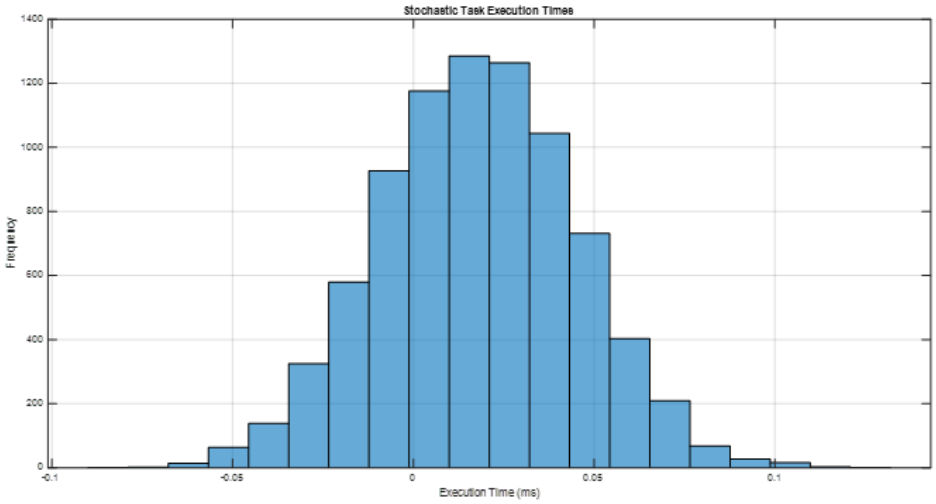


Figure 8. Stochastic task execution times.

The stochastic nature of task execution times, depicted in the Gaussian distribution, highlights the unpredictability of task durations in distributed environments. This randomness impacts system stability and resource utilization, emphasizing the need for robust scheduling algorithms that can accommodate such variations.

Impact on Markov Model and Scheduling Strategies:

By incorporating dynamic arrival rates, resource heterogeneity, and stochastic execution times, our enhanced Markov model provides a comprehensive framework for analyzing scheduling and offloading strategies in cloud, fog, and edge computing environments. These enhancements allow the model to do the following:

- Adapt to Dynamic Workloads: The variability in task arrivals and execution times drives state transitions, enabling the model to simulate how systems respond to fluctuating workloads and adapt their scheduling decisions accordingly.
- Optimize Resource Utilization: The heterogeneous nature of resources guides the scheduling algorithms to allocate tasks to the most suitable nodes, improving overall system efficiency and minimizing energy consumption.
- Handle Unpredictability: Stochastic elements introduce realistic execution variability, helping identify which scheduling techniques are most resilient under uncertain conditions.

- The integration of these dynamic and stochastic elements significantly advances the Markov model's ability to capture the complex behaviors of distributed computing systems. The results demonstrate the effectiveness of adaptive, resource-aware scheduling strategies that can maintain performance under varying conditions, validating the model's practical applicability in the real world.

4.6. Modeling Scheduling Decisions

Scheduling decisions are a critical component of our enhanced model, directly influencing the performance of cloud, fog, and edge layers. The allocation of tasks is based on real-time assessments of system states, which consider current load, resource availability, and task priorities. This dynamic decision-making process ensures that the model adapts effectively to changing conditions, maintaining optimal performance across all layers.

- Real-Time State Assessment: At each time step, the system evaluates the current status of all nodes. This assessment includes monitoring current load levels, resource availability, and task priorities, which are critical for determining the best allocation strategy. For example, if edge resources are near maximum capacity, the system dynamically adjusts to offload tasks to fog or cloud layers, prioritizing tasks that are less sensitive to latency.
- Task Allocation: Tasks are assigned to the most appropriate layer (cloud, fog, or edge) based on their specific requirements. Latency-sensitive tasks are prioritized for edge or fog processing, ensuring rapid response times. Tasks that require significant computational resources but have flexible latency requirements are directed to the cloud. This allocation strategy is dynamically adjusted based on the current system state, optimizing resource use and minimizing delays.
- Feedback Loops: The model incorporates feedback mechanisms where the outcomes of previous scheduling decisions influence future allocations. For instance, if a prior decision to allocate a task to the cloud resulted in a significant delay, the model adapts by favoring fog or edge layers for similar tasks in subsequent steps. This feedback loop ensures that the system continuously learns from its past performance, refining its decision-making process to better handle varying workloads.

The incorporation of real-time state assessments, adaptive task allocation, and feedback loops significantly enhances the model's scheduling capabilities. These elements directly influence the performance metrics observed in Figures 3–6, demonstrating the effectiveness of adaptive decision-making in distributed environments. By dynamically allocating tasks based on real-time conditions, the system optimizes response times, improves resource utilization, and reduces energy consumption.

The dynamic nature of the scheduling decisions modeled in this approach reflects the real-world complexity of distributed computing environments. This adaptability is crucial for applications in smart cities, healthcare, and other sectors where task priorities and system loads can change rapidly. By incorporating these dynamic scheduling decisions into the Markov model, the enhanced framework provides a realistic and adaptable approach to managing distributed applications. The model's ability to assess real-time conditions, allocate tasks appropriately, and learn from feedback ensures optimal performance under varying conditions, validating its use in complex, real-world scenarios.

To ensure the robustness of our chosen approach, we conducted a thorough comparison of our custom Markov-based model with two widely recognized simulators: CloudSim v6.0.0-beta and iFogSim2. This comparison, summarized in Table 6, highlights the unique advantages of our model in the context of distributed application scheduling and offloading across cloud, fog, and edge computing environments. The results underscore the impor-

tance of using a tailored simulation approach that can capture the complex dynamics of these environments.

Table 6. Comparison of modeling approaches.

Feature	CloudSim	iFogSim	Custom Markov Model
Cloud Computing Support	High	High	High
Fog/Edge Computing Support	Low	High	High
Dynamic Task Arrival Rates	Medium	High	High
Resource Heterogeneity	Low	Medium	High
Stochastic Execution Times	Low	Medium	High
Scheduling Flexibility	Medium	Medium	High
Customization Ability	Low	Medium	High

CloudSim is a well-established tool that excels in simulating cloud environments, particularly in terms of virtual machine management and resource provisioning. It provides a solid framework for cloud computing simulations, allowing researchers to model resource allocation, load balancing, and other cloud-specific tasks effectively. However, it has limitations when it comes to modeling fog and edge computing environments, which require more detailed handling of latency-sensitive and resource-constrained tasks. While CloudSim provides moderate support for dynamic task arrival rates and resource heterogeneity, it falls short in areas such as stochastic execution times and scheduling flexibility, which are critical for accurately simulating real-world distributed systems.

iFogSim, an extension of CloudSim, addresses some of these limitations by offering improved support for fog and edge computing. It is particularly effective for modeling scenarios involving mobility, microservices, and IoT-based applications, making it a more versatile tool for simulations that extend beyond traditional cloud environments. However, even iFogSim has its constraints; fully integrating stochastic elements and achieving the level of customization required for our study would necessitate significant adaptation. The integration of these features often requires complex modifications to the underlying framework, making iFogSim less adaptable for studies that require highly specific and dynamic modeling.

Custom Markov-Based Model: In contrast, our custom Markov-based model was specifically designed to overcome these limitations. It offers high flexibility and adaptability, allowing for the incorporation of dynamic task arrival rates, resource heterogeneity, and stochastic execution times—features that are essential for accurately simulating complex, real-world distributed systems. This model not only supports dynamic scheduling decisions but also provides a more detailed and realistic understanding of system performance across cloud, fog, and edge layers. Its unique ability to model non-linear and unpredictable behaviors gives it an edge over existing simulators, particularly in scenarios that demand high levels of customization and precision.

4.7. Comparison with Advanced Simulators

To ensure the robustness of our chosen approach, we conducted a thorough comparison of our custom Markov-based model with two widely recognized simulators: CloudSim and iFogSim. This comparison, summarized in Table 6, highlights the unique advantages of our model in the context of distributed application scheduling and offloading across cloud, fog, and edge computing environments.

Indeed, CloudSim is a well-established tool that excels in simulating cloud environments, particularly in terms of virtual machine management and resource provisioning.

However, it has limitations when it comes to modeling fog and edge computing environments, which require more detailed handling of latency-sensitive and resource-constrained tasks. While CloudSim provides moderate support for dynamic task arrival rates and resource heterogeneity, it falls short in areas such as stochastic execution times and scheduling flexibility.

iFogSim, an extension of CloudSim, addresses some of these limitations by offering improved support for fog and edge computing. It is particularly effective for modeling scenarios that involve mobility, microservices, and IoT-based applications. However, even iFogSim has its constraints; fully integrating stochastic elements and achieving the level of customization required for our study would necessitate significant adaptation.

In contrast, our custom Markov-based model was specifically designed to overcome these limitations. It offers high flexibility and adaptability, allowing for the incorporation of dynamic task arrival rates, resource heterogeneity, and stochastic execution times—features that are essential for accurately simulating complex, real-world distributed systems. This model not only supports dynamic scheduling decisions but also provides a more detailed and realistic understanding of system performance across cloud, fog, and edge layers.

Table 6 highlights the strengths and weaknesses of each modeling approach. While CloudSim and iFogSim provide solid frameworks for simulation, their limitations in handling the complexities of fog and edge computing environments underscore the need for a more tailored solution. Our custom Markov-based model addresses these gaps by offering a nuanced and adaptable simulation environment, which is crucial for the in-depth analysis of scheduling and offloading strategies. The model's ability to capture non-linear behaviors and provide accurate predictions of system utilization, response times, and energy efficiency makes it particularly valuable for designing efficient and reliable distributed systems.

Cloud Computing Support: All three models provide strong support for cloud computing, but only iFogSim and the custom Markov model extend this support to fog and edge environments, crucial for latency-sensitive and resource-constrained tasks.

Dynamic Task Arrival Rates and Stochastic Execution Times: The custom Markov model stands out by fully integrating dynamic task arrival rates and stochastic execution times. This level of integration is critical for capturing real-world variability and adapting scheduling decisions dynamically.

Resource Heterogeneity and Scheduling Flexibility: The custom Markov model provides high adaptability and flexibility, outperforming both CloudSim and iFogSim. This capability allows it to handle diverse resource types and optimize task allocation more effectively.

Customization Ability: The custom Markov-based approach offers unparalleled customization, allowing researchers to tailor the model precisely to their study's requirements. This makes it particularly valuable for in-depth analysis of distributed application scheduling and offloading.

Overall, the comparison in Table 6 clearly demonstrates that while CloudSim and iFogSim have their merits, our custom Markov-based model provides a superior framework for the specific requirements of this study. By integrating advanced features such as stochastic execution times, dynamic task arrival rates, and high customization ability, the Markov-based approach enables a deeper and more realistic exploration of distributed application scheduling and offloading. This tailored solution allows for accurate predictions of system utilization, response times, and energy efficiency, ultimately leading to more effective and practical strategies for designing efficient and reliable distributed systems.

5. Conclusions

This research has presented a comprehensive examination of distributed application scheduling and offloading in cloud, fog, and edge computing environments. By leveraging an enhanced Markov-based model and employing MATLAB to compute steady-state probabilities, we have effectively analyzed the dynamic behavior, resource utilization, and stability of these systems under varying conditions. Our results demonstrate that integrating dynamic scheduling algorithms and offloading strategies significantly improves key performance metrics, including network load, response delay, energy consumption, and operational costs.

The enhanced Markov-based model plays a pivotal role in capturing the complexities of distributed systems. By incorporating dynamic task arrival rates, resource heterogeneity, and stochastic execution times, the model offers a more nuanced simulation of real-world scenarios. In comparison to traditional simulators like CloudSim and iFogSim, our approach provides superior flexibility and predictive accuracy, particularly in handling the stochastic and dynamic nature of workloads across cloud, fog, and edge layers.

This study's key contributions include the implementation and comparative analysis of scheduling and offloading techniques, such as priority-based scheduling, load balancing, resource-aware scheduling, partial and full offloading, and cooperative offloading. These techniques collectively enhance system performance, optimize resource utilization, and reduce latency. The model's ability to capture non-linear and unpredictable system behaviors has proven invaluable for designing robust scheduling strategies capable of adapting to the challenges of dynamic distributed environments.

The findings further highlight the critical role of fog and edge computing in reducing response times by processing tasks closer to the data source, making these paradigms particularly effective for real-time applications. Additionally, they demonstrate significant reductions in operational costs and energy consumption, advancing sustainable computing practices. The model's adaptive nature ensures scalability and resilience, dynamically allocating resources to accommodate varying workloads, underscoring the practicality and robustness of the proposed strategies. Future research could build upon these findings by exploring advanced scheduling algorithms that leverage artificial intelligence, addressing emerging security and privacy concerns, and validating the proposed techniques through large-scale real-world deployments. Incorporating more intricate stochastic models, real-time feedback mechanisms, and multi-tier optimization strategies could further refine the system's predictive capabilities.

In conclusion, this study underscores the transformative potential of distributed application scheduling and offloading across cloud, fog, and edge environments. The proposed methodologies pave the way for more adaptable, efficient, and sustainable computing systems, meeting the evolving demands of modern applications and advancing the field of distributed systems.

Funding: The author extends his appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number "NBU-FPEJ-2025-2429-01".

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Yi, S.; Li, C.; Li, Q. A survey of fog computing: Concepts, applications and issues. In Proceedings of the 2015 Workshop on Mobile Big Data, Hangzhou, China, 21 June 2015; pp. 37–42.
2. Kraemer, F.A.; Braten, A.E.; Tamkittikhun, N.; Palma, D. Fog computing in healthcare—A review and discussion. *IEEE Access* **2017**, *5*, 9206–9222. [CrossRef]
3. Habibi, P.; Farhoudi, M.; Kazemian, S.; Khorsandi, S.; Leon-Garcia, A. Fog computing: A comprehensive architectural survey. *IEEE Access* **2020**, *8*, 69105–69133. [CrossRef]
4. Das, R.; Inuwa, M.M. A review on fog computing: Issues, characteristics, challenges, and potential applications. *Telemat. Inform. Rep.* **2023**, *10*, 100049. [CrossRef]
5. Costa, B.; Bachiega, J., Jr.; De Carvalho, L.R.; Araujo, A.P. Orchestration in fog computing: A comprehensive survey. *ACM Comput. Surv. (CSUR)* **2022**, *55*, 1–34. [CrossRef]
6. Duan, Q.; Wang, S.; Ansari, N. Convergence of networking and cloud/edge computing: Status, challenges, and opportunities. *IEEE Netw.* **2020**, *34*, 148–155. [CrossRef]
7. Ali, B.; Gregory, M.A.; Li, S. Multi-access edge computing architecture, data security and privacy: A review. *IEEE Access* **2021**, *9*, 18706–18721. [CrossRef]
8. Mansouri, Y.; Babar, M.A. A review of edge computing: Features and resource virtualization. *J. Parallel Distrib. Comput.* **2021**, *150*, 155–183. [CrossRef]
9. Wang, Y.; Yang, C.; Lan, S.; Zhu, L.; Zhang, Y. End-edge-cloud collaborative computing for deep learning: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2024**, *26*, 2647–2683. [CrossRef]
10. Silitonga, D.; Rohmayanti, S.A.A.; Aripin, Z.; Kuswandi, D.; Sulistyo, A.B. Edge computing in E-commerce business: Economic impacts and advantages of scalable information systems. *EAI Endorsed Trans. Scalable Inf. Syst.* **2023**, *11*. [CrossRef]
11. Quy, N.M.; Ngoc, L.A.; Ban, N.T.; Hau, N.V.; Quy, V.K. Edge computing for real-time Internet of Things applications: Future internet revolution. *Wirel. Pers. Commun.* **2023**, *132*, 1423–1452. [CrossRef]
12. Asghari, A.; Sohrabi, M.K. Server placement in mobile cloud computing: A comprehensive survey for edge computing, fog computing and cloudlet. *Comput. Sci. Rev.* **2024**, *51*, 100616. [CrossRef]
13. Tank, B.; Gandhi, V. A Comparative Study on Cloud Computing, Edge Computing and Fog Computing. In *Recent Developments in Electronics and Communication Systems*; IOS Press: Amsterdam, The Netherlands, 2023; pp. 665–670.
14. Hunko, M.; Tkachov, V.; Kovalenko, A.; Kuchuk, H. Advantages of fog computing: A comparative analysis with cloud computing for enhanced edge computing capabilities. In Proceedings of the 2023 IEEE 4th KhPI Week on Advanced Technology (KhPIWeek), Kharkiv, Ukraine, 2–6 October 2023; pp. 1–5.
15. Salaht, F.A.; Desprez, F.; Lebre, A. An overview of service placement problem in fog and edge computing. *ACM Comput. Surv. (CSUR)* **2020**, *53*, 1–35. [CrossRef]
16. Kaur, G.; Batth, R.S. Edge computing: Classification, applications, and challenges. In Proceedings of the 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM), London, UK, 28–30 April 2021; pp. 254–259.
17. Laroui, M.; Nour, B.; Moungla, H.; Cherif, M.A.; Afifi, H.; Guizani, M. Edge and fog computing for IoT: A survey on current research activities & future directions. *Comput. Commun.* **2021**, *180*, 210–231.
18. Simić, M.; Prokić, I.; Dedeić, J.; Sladić, G.; Milosavljević, B. Towards edge computing as a service: Dynamic formation of the micro data-centers. *IEEE Access* **2021**, *9*, 114468–114484. [CrossRef]
19. Sheikh Sofla, M.; Haghi Kashani, M.; Mahdipour, E.; Faghih Mirzaee, R. Towards effective offloading mechanisms in fog computing. *Multimed. Tools Appl.* **2022**, *81*, 1997–2042. [CrossRef]
20. Zhang, M.; Peng, Y.; Zhu, J.; Yin, Q. Efficient flow-based scheduling for geo-distributed simulation tasks in collaborative edge and cloud environments. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 3442–3459.
21. Alsaffar, Q.S.; Ayed, L.B. Heterogeneous Resources in Infrastructures of the Edge Network Paradigm: A Comprehensive Review. *Karbala Int. J. Mod. Sci.* **2024**, *10*, 15. [CrossRef]
22. Ebrahim, M.; Hafid, A. Fully Distributed Fog Load Balancing with Multi-Agent Reinforcement Learning. *arXiv* **2024**, arXiv:2405.12236.
23. Altowaijri, S.M. Workflow Scheduling and Offloading for Service-based Applications in Hybrid Fog-Cloud Computing. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 726–735. [CrossRef]
24. Fayos Jordan, R. Edge-Fog-Cloud Computing System Analysis and Implementations for IoT and Smart Cities. Available online: https://hdl.handle.net/10550/99404 (accessed on 15 September 2024).
25. Bosaeed, S.; Katib, I.; Mehmood, R. A Fog-Augmented Machine Learning based SMS Spam Detection and Classification System. In Proceedings of the 2020 5th International Conference on Fog and Mobile Edge Computing (FMEC), Paris, France, 20–23 April 2020; pp. 325–330.

26. Wang, K.; Kong, S.; Chen, X.; Zhao, M. Edge Computing Empowered Smart Healthcare: Monitoring and Diagnosis with Deep Learning Methods. *J. Grid Comput.* **2024**, *22*, 30. [CrossRef]
27. Khanh, Q.V.; Nguyen, V.H.; Minh, Q.N.; Van, A.D.; Le Anh, N.; Chehri, A. An efficient edge computing management mechanism for sustainable smart cities. *Sustain. Comput. Inform. Syst.* **2023**, *38*, 100867. [CrossRef]
28. Abu-Amssimir, N.; Al-Haj, A. A QoS-aware resource management scheme over fog computing infrastructures in IoT systems. *Multimed. Tools Appl.* **2023**, *82*, 28281–28300. [CrossRef]
29. Salis, A.; Jensen, J. A smart fog-to-cloud system in airport: Challenges and lessons learnt. In Proceedings of the 2020 21st IEEE International Conference on Mobile Data Management (MDM), Versailles, France, 30 June–3 July 2020; pp. 359–364.
30. Janbi, N.; Katib, I.; Albeshri, A.; Mehmood, R. Distributed artificial intelligence-as-a-service (DAIaaS) for smarter IoE and 6G environments. *Sensors* **2020**, *20*, 5796. [CrossRef] [PubMed]
31. Hazra, A.; Adhikari, M.; Amgoth, T.; Srirama, S.N. Joint computation offloading and scheduling optimization of IoT applications in fog networks. *IEEE Trans. Netw. Sci. Eng.* **2020**, *7*, 3266–3278. [CrossRef]
32. Okegbile, S.D.; Maharaj, B.T.; Alfa, A.S. A multi-user tasks offloading scheme for integrated edge-fog-cloud computing environments. *IEEE Trans. Veh. Technol.* **2022**, *71*, 7487–7502. [CrossRef]
33. Rahimikhanghah, A.; Tajkey, M.; Rezazadeh, B.; Rahmani, A.M. Resource scheduling methods in cloud and fog computing environments: A systematic literature review. *Clust. Comput.* **2022**, *25*, 911–945. [CrossRef]
34. Abdulazeez, D.H.; Askar, S.K. Offloading mechanisms based on reinforcement learning and deep learning algorithms in the fog computing environment. *IEEE Access* **2023**, *11*, 12555–12586. [CrossRef]
35. Ahmed, O.H.; Lu, J.; Ahmed, A.M.; Rahmani, A.M.; Hosseinzadeh, M.; Masdari, M. Scheduling of scientific workflows in multi-fog environments using Markov models and a hybrid salp swarm algorithm. *IEEE Access* **2020**, *8*, 189404–189422. [CrossRef]
36. Javaheri, D.; Gorgin, S.; Lee, J.A.; Masdari, M. An improved discrete harris hawk optimization algorithm for efficient workflow scheduling in multi-fog computing. *Sustain. Comput. Inform. Syst.* **2022**, *36*, 100787. [CrossRef]
37. Tran-Dang, H.; Kim, D.S. Dynamic collaborative task offloading for delay minimization in the heterogeneous fog computing systems. *J. Commun. Netw.* **2023**, *25*, 244–252. [CrossRef]
38. Hussein, M.K.; Mousa, M.H. Efficient task offloading for IoT-based applications in fog computing using ant colony optimization. *IEEE Access* **2020**, *8*, 37191–37201. [CrossRef]
39. Siar, H.; Izadi, M. Offloading coalition formation for scheduling scientific workflow ensembles in fog environments. *J. Grid Comput.* **2021**, *19*, 34. [CrossRef]
40. Bansal, S.; Aggarwal, H.; Aggarwal, M. A systematic review of task scheduling approaches in fog computing. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e4523. [CrossRef]
41. Zhang, P.; Chen, N.; Xu, G.; Kumar, N.; Barnawi, A.; Guizani, M.; Yu, K. Multi-target-aware dynamic resource scheduling for cloud-fog-edge multi-tier computing network. *IEEE Trans. Intell. Transp. Syst.* **2023**, *25*, 3885–3897. [CrossRef]
42. Wang, Z.; Goudarzi, M.; Gong, M.; Buyya, R. Deep reinforcement learning-based scheduling for optimizing system load and response time in edge and fog computing environments. *Future Gener. Comput. Syst.* **2024**, *152*, 55–69. [CrossRef]
43. Sharma, N.; Bhatt, R. FoG Computing based IoT in Healthcare Application. In Proceedings of the 2020 Sixth International Conference on Parallel Distributed and Grid Computing (PDGC), Waknaghat, India, 6–8 November 2020; pp. 442–446.
44. Yang, G.; Hou, L.; He, X.; He, D.; Chan, S.; Guizani, M. Offloading time optimization via Markov decision process in mobile-edge computing. *IEEE Internet Things J.* **2020**, *8*, 2483–2493. [CrossRef]
45. Mukherjee, M.; Matam, R.; Shu, L.; Maglaras, L.; Ferrag, M.A.; Choudhury, N.; Kumar, V. Security and privacy in fog computing: Challenges. *IEEE Access* **2017**, *5*, 19293–19304. [CrossRef]
46. Zippo, R.; Stea, G. Computationally efficient worst-case analysis of flow-controlled networks with network calculus. *IEEE Trans. Inf. Theory* **2023**, *69*, 2664–2690. [CrossRef]
47. Andreoli, R.; Zhao, J.; Cucinotta, T.; Buyya, R. CloudSim 7G: An Integrated Toolkit for Modeling and Simulation of Future Generation Cloud Computing Environments. *arXiv* **2024**, arXiv:2408.13386. [CrossRef]
48. Mansouri, N.; Ghafari, R.; Zade, B.M.H. Cloud computing simulators: A comprehensive review. *Simul. Model. Pract. Theory* **2020**, *104*, 102144. [CrossRef]
49. Koot, M.; Wijnhoven, F. Usage impact on data center electricity needs: A system dynamic forecasting model. *Appl. Energy* **2021**, *291*, 116798. [CrossRef]
50. Kong, X.; Wu, Y.; Wang, H.; Xia, F. Edge computing for internet of everything: A survey. *IEEE Internet Things J.* **2022**, *9*, 23472–23485. [CrossRef]
51. Haibeh, L.A.; Yagoub, M.C.; Jarray, A. A survey on mobile edge computing infrastructure: Design, resource management, and optimization approaches. *IEEE Access* **2022**, *10*, 27591–27610. [CrossRef]

52. Mahmud, R.; Pallewatta, S.; Goudarzi, M.; Buyya, R. Ifogsim2: An extended ifogsim simulator for mobility, clustering, and microservice management in edge and fog computing environments. *J. Syst. Softw.* **2022**, *190*, 111351. [CrossRef]
53. Available online: https://www.kaggle.com/datasets/gauravdhamane/gwa-bitbrains (accessed on 9 August 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.