Journal of Internet Services and Applications , 2026, 17:1, doi: 10.5753/jisa.2026.6851

Stochastic Petri Nets for Drone Performance Analysis in Mobile Edge Computing

Iago Roberto Almeida [**Federal University of Piauí** | ***iago.almeida@ufpi.edu.br***]
José Miqueias Araújo [**Federal University of Piauí** | ***jmiqueias@ufpi.edu.br***]
Gustavo Callou [**Federal Rural University of Pernambuco** | ***gustavo.callou@ufrpe.br***]
Francisco Airton Silva [**Federal University of Piauí** | ***faps@ufpi.edu.br***]

Laboratory of Applied Research to Distributed Systems (PASID), Federal University of Piauí (UFPI), Picos, Piauí, Brazil; R. Cicero Duarte, No. 905 - Junco, 64607-670

Received: 10 November 2025 • **Accepted:** 22 May 2026 • **Published:** 25 June 2026

Abstract The global drone market has grown from US$4 billion in 2023 to a projected US$4.8 billion by 2029. Concomitant with this growth, integrating drones into Mobile Edge Computing (MEC) environments poses critical challenges in scalability, resource allocation, and latency-aware distributed processing, which directly affect overall system performance. Addressing these issues through physical prototyping is costly and complex, which motivates the use of analytical models that can predict system behavior under different conditions. In this work, we propose a performance evaluation approach based on Stochastic Petri Nets (SPNs) to model the admission, load balancing, and distributed task processing among drones acting as mobile edge nodes. The proposed model enables resource scaling and scalability analysis without physically implementing the architecture, reducing deployment risks and costs. Simulation results demonstrate that the model accurately captures key performance metrics, including Mean Response Time (MRT), throughput, and task drop rate, providing quantitative insights for designing efficient and resilient UAV-supported MEC systems.

Keywords: Mobile Edge Computing (MEC), Unmanned Aerial Vehicles (UAVs), Performance Evaluation, Stochastic Petri Nets

1 Introduction

Between 2018 and 2023, the global drone market experienced significant growth, reaching approximately four billion dollars in 2023, with projections pointing to $4.8 billion by 2029 Statista [2023]. This growth reflects the expanding use of drones in critical applications such as security, industrial inspection, environmental monitoring, autonomous deliveries, geospatial mapping, and emergency response Hossain [2022]. With technological advancements, these devices have evolved to incorporate advanced sensors and embedded modules, performing a dual role: collecting data and performing local processing. This characteristic aligns them with the Mobile Edge Computing (MEC) paradigm, in which computing resources are distributed at the network edge, close to data-generating sources, to reduce latency and increase efficiency in distributed contexts Rojas-Perez and Martínez-Carranza [2021].

The MEC architecture enables drones to act as mobile and highly dynamic edge nodes, capable of performing decentralized processing while simultaneously cooperating with other network elements to optimize resource utilization. Unlike the traditional cloud-based model, which relies heavily on stable connectivity and bandwidth, MEC allows critical real-time analysis to be performed directly at the edge, reducing the need to continuously transmit large volumes of data Emimi *et al*. [2023]; Park *et al*. [2021]. Such edge-based processing is significant for applications that require low latency, including visual recognition, drone fleet coordination, and event detection in rapid-response scenarios. Furthermore, MEC contributes to load balancing and increased resilience, as drones can make autonomous decisions without relying entirely on centralized infrastructure Bartolini *et al*. [2020].

Despite these advances, consolidating drones into the MEC ecosystem remains a significant challenge. Notable among these are limited onboard hardware capacity, inefficiencies in resource allocation, and latency-aware distributed processing demands in dynamic environments. These factors directly affect the scalability and performance of applications that rely on real-time processing and distributed decision-making Shnaiwer *et al*. [2022]; Abubakar *et al*. [2023]. Therefore, it is essential to have quantitative modeling and analysis methods that allow for systematic assessment of the impact of these constraints on overall system performance. This provides a structured basis for architectural design and capacity planning decisions in UAV-assisted MEC deployments.

The proposed SPN-based framework focuses on the operational and computational aspects of the UAV-MEC architecture, capturing the stochastic behavior of task admission, load balancing, and distributed processing among drones acting as edge nodes. Aspects such as wireless channel modeling, drone mobility, energy consumption, and battery constraints are not explicitly incorporated into the current model. Although these are recognized as relevant factors in real-world UAV-assisted deployments, their inclusion would significantly increase the complexity of the SPN representation and shift the analytical focus away from the performance evaluation objectives of this work Ai *et al*. [2025]; Jeong *et al*. [2017]. These aspects are therefore considered important

directions for future extensions of the proposed work.

Based on this context, this paper proposes an approach based on Stochastic Petri Nets (SPNs) to model and evaluate the performance of drone systems integrated with the MEC architecture. SPNs constitute mathematical and graphical models suitable for representing concurrent and stochastic systems, enabling the analysis of aspects such as synchronization, queuing, and temporal variations Marsan *et al*. [1998]. The proposal enables the exploration of the system's dynamic behavior under various operational conditions and the evaluation of metrics such as MRT, loss probability, processing rate, and resource utilization. In addition, the work includes transient analysis using absorbing models and the application of experimental design (DoE) techniques, providing a quantitative basis for planning and optimizing MEC systems based on intelligent drones. The main contributions of this paper include:

- **Non-Absorbent SPN Model:** An SPN model developed for steady-state analysis, allowing for continuous performance evaluation of MEC systems with drones as edge nodes, considering metrics such as MRT, throughput, packet loss rate, and computational resource utilization.
- **Absorbent SPN Model:** An extension of the SPN model focused on transient analysis, designed to evaluate the total processing time of finite batches of tasks. This model enables the generation of cumulative distribution functions (CDFs) for request completion times, making it useful for simulating controlled loads or MEC applications with batched tasks.
- **Sensitivity Analysis with DoE:** A complete factorial design of experiments approach used to identify the most influential factors on system performance, especially on MRT. This analysis helps understand the interactions among variables and fine-tune MEC system configurations.

This paper is organized as follows: Section 2 presents the main work related to the proposal. Section 3 details the system's functional architecture and its components. Section 4 describes the non-absorbing SPN model and its main characteristics. Section 5 presents two case study scenarios. Section 6 presents the absorbing SPN model, focusing on generating and analyzing CDFs. Section 7 discusses the results of the sensitivity analysis performed with DoE. Finally, Section 8 presents the concluding remarks of the work.

2 Related Works

This section presents a literature review on the use of the MEC architecture and drones. The research highlights a range of methodological approaches, from optimization-based analytical models to Petri net modeling techniques. The main objectives include reducing energy consumption, improving system availability, and ensuring real-time quality of service. In this context, each study offers specific solutions to challenges such as scalability, reliability, and distributed processing, providing important insights for advancing drone-based systems. Table 1 presents a comparison of the studies collected in this paper.

Fedorova *et al*. [2022] proposes a model based on Colored Petri Nets (CPNs) to evaluate drone inspection methods in electrical distribution networks. The modular model considers typical UAV (Unmanned Aerial Vehicle) system configuration elements, drone types, and application patterns, allowing for estimation of monitoring frequency, optimal UAV grouping, and maintenance periods. A key advantage is the inclusion of stochastic processes, such as failures and maintenance, which facilitates the conceptual design of monitoring systems. The work supports everyday flight tasks and optimizes costs while maintaining the required monitoring quality. However, a relevant limitation of this methodology is that it does not account for external stochastic factors, such as weather conditions, which can significantly impact flight monitoring and operational performance.

Zeng and Tang [2023] investigates real-time data acquisition and processing assisted by MEC for general UAV missions, defining a theoretical data acquisition rate and real-time processing quality (QoR). They minimize energy consumption and mission time by optimizing trajectory, resource allocation, and duration, using successive convex approximation and block-based coordinate descent. The focus on non-contact Data Acquisition (DA) differs from passive collections, emphasizing dynamic mobility and real-time analysis for smart applications such as power grids. The algorithm derives closed-loop solutions for transmission and computation, demonstrating significant performance gains. Despite these contributions, the proposed system is limited by its assumption of simplified flight dynamics, such as constant altitude. It relies on basic communication assumptions, failing to address more practical and complex channel models, including probabilistic Line-of-Sight (LoS) and Non-LoS scenarios.

Urvashi and Bansal [2025] uses Petri Net (PN) modeling and a PSO (Particle Swarm Optimization) algorithm to improve the availability of a wireless drone sensor system for fire extinguishers (FEWSD). The PN simulation analyzes critical subsystems, prioritizes maintenance, assesses the impact of parameters on system availability, and determines the necessary repair personnel. The PSO optimizes failure and repair rates, increasing availability from 0.9883 to 0.9937, with an emphasis on the sensor subsystem. The work addresses limitations of traditional fire detection and suppression systems, especially in remote areas. However, a significant limitation of this study is the assumption of constant failure and repair rates across all subsystems, which may not reflect real-world scenarios in which these rates are often time-dependent or influenced by environmental degradation. Furthermore, the model lacks a detailed analysis of the communication overhead and energy constraints associated with the PSO execution in a resource-constrained drone environment.

Zhang *et al*. [2019] addresses stochastic offloading and trajectory planning in UAV-assisted MEC to minimize the average weighted energy, subject to task queues, resource allocation, and mobility. Using the Lyapunov approach, they decompose the problem into subproblems that are solved iteratively, considering infrastructure-deficient scenarios such as disasters. The temporal capture and non-convexity model outperforms benchmarks in terms of energy consumption and computational rates. Despite these gains, a key limitation of this work is that the proposed trajectory scheduling assumes

Table 1. Related Works Comparison

Work	Evaluation Method	Metrics	MEC with Drones	SPN-based Modeling
Fedorova *et al*. [2022]	Simulation with CPNs	Monitoring frequency, cost, flight time, maintenance periods	✗	✗
Zeng and Tang [2023]	Optimization methods	Energy consumption, mission time, processing QoR	✓	✗
Urvashi and Bansal [2025]	PN + PSO simulation	System availability, maintenance priority	✗	✗
Zhang *et al*. [2019]	Lyapunov + optimization	Weighted energy, computation rate	✓	✗
Luo *et al*. [2021]	Blockchain-based simulation	Latency, auditability	✓	✗
Miao *et al*. [2023]	Path planning + optimization	Total latency, energy efficiency	✓	✗
Liu *et al*. [2022]	Convex approximation	Secrecy rate, task completion time	✓	✗
Carvalho *et al*. [2020]	Analytical SPN modeling	MRT, resource utilization	✗	✓
Liu *et al*. [2025]	MADRL-based simulation	System latency, energy consumption	✗	✗
Sun *et al*. [2025]	Experimental evaluation	Inference latency, energy consumption	✗	✗
This work	SPN modeling	MRT, throughput, drop rate, utilization	✓	✓

a fixed flight altitude to maintain line-of-sight, which may be impractical in dense urban or obstacle-rich environments. Additionally, the Lyapunov-based decomposition requires careful tuning of the tradeoff parameter (V), and the study does not fully explore the impact of extreme channel fluctuations or intermittent connectivity on the stability of the task queues.

Luo *et al*. [2021] proposes an architecture for offloading MEC tasks using drones, employing a private blockchain to ensure visibility and auditability for MEC providers, thus extending coverage to rural or emergency areas. Drones cache IoT data and forward it to MEC servers, selecting the best offloading target through smart contract policies designed to reduce task transmission and computation delays. The permissioned network ensures authority and an immutable processing path. However, a significant limitation of this approach is its reliance on a private blockchain, which limits the system's scalability to a small number of MEC servers to maintain a viable consensus time. Furthermore, because it relies solely on simulated experiments, the model may not adequately capture the complex physical and energy constraints of real-world drone flight dynamics, particularly when assuming a homogeneous server network.

Miao *et al*. [2023] develops trajectory planning algorithms for drone swarms in MEC for IoT, using a global scheduling strategy that prioritizes area, residual energy, and distance to minimize flight length and energy consumption. They optimize local communication coverage and offloading while accounting for user mobility, maximizing service availability and minimizing total latency, thereby achieving superior energy efficiency in complex scenarios. Despite these contributions, a critical problem in this study is the strong reliance on a centralized ground base station for global route planning control, which introduces a single point of failure and potential communication bottlenecks in severely dynamic or disaster environments. Furthermore, the assumption of simplified mathematical models, such as equal bandwidth allocation among accessed devices, may not reflect the heterogeneous and unpredictable nature of traffic in real IoT networks.

Liu *et al*. [2022] formulates a secure Mobile Edge Computing (MEC) framework with Drone Base Stations (DBSs) using Free Space Optics (FSO) for backhauling to a Macro Base Station (MBS), optimizing bandwidth/computing allocation, power, User Equipment (UE) association, and DBS placement to maximize the secrecy rate and minimize task time. They decompose these complex, mixed-integer, nonlinear problems into subproblems solved iteratively via successive convex approximation, improving the secrecy rate by 19% over baselines. However, a notable limitation of this approach is the reliance on FSO for the backhaul link, which, while offering high capacity, is highly susceptible to atmospheric conditions (such as fog, rain, or turbulence) and requires strict Line-of-Sight (LoS) alignment, potentially reducing reliability in dynamic drone environments or adverse weather. Additionally, the iterative decomposition of the non-convex problem may lead to sub-optimal local solutions and introduce high computational overhead, making real-time execution challenging.

Carvalho *et al*. [2020] proposes a Stochastic Petri Net (SPN) model to evaluate MEC performance in metropolitan areas, focusing on the essential tradeoff between Mean Response Time (MRT) and server resource utilization. The model analyzes the dynamic allocation of requests near users and is validated through numerical analyses using real-world values, serving as a practical guide for administrators in architectural adaptations to minimize latency and optimize the use of MEC servers. Despite its practical applicability, a primary limitation of using SPN models in large-scale MEC environments is the "state-space explosion" problem, which makes analytical modeling computationally intractable as the number of mobile users, MEC servers, and queue capacities increases. Furthermore, the model relies on generalized statistical distributions and may not adequately capture users' highly dynamic, unpredictable mobility patterns in real-time metropolitan scenarios.

Liu *et al*. [2025] proposed a cooperative task offloading framework for Mobile Edge Computing (MEC) in future networks, employing a Multi-Agent Deep Reinforcement Learning (MADRL) approach. Evaluated through simulation, their method successfully aims to minimize system latency and energy consumption for edge devices. However, their architecture primarily targets traditional terrestrial edge nodes. It does not address the highly dynamic topology, extreme energy constraints, and specific hardware constraints inherent to Unmanned Aerial Vehicles (UAVs) operating as edge servers. Moreover, relying exclusively on DRL-based simulation limits the ability to perform rigorous mathematical verification and to predict analytical scalability, which are essential to guarantee system reliability under distinct stochastic load conditions.

Sun *et al*. [2025] introduced Intra-DP, a collaborative inference system specifically designed to optimize Deep Neural Network (DNN) task partitioning in MEC environments. Their experimental evaluation demonstrated practical improvements in inference latency and battery energy consumption using real-world hardware deployments. Despite these practical benefits, their work focuses primarily on general-

purpose mobile devices and fails to model the complex communication infrastructure needed to integrate drones as mobile edge nodes. Furthermore, the absence of a formal analytical modeling strategy restricts comprehensive capacity planning, admission control evaluation, and performance prediction when the system faces connectivity uncertainties.

This work presents a SPN–based performance evaluation framework that models drones as mobile edge nodes within a MEC architecture, combining both a non-absorbing model for steady-state analysis and an absorbing variant for transient (batch) analysis; it quantifies MRT, throughput, task drop rate and resource utilization, and uses DoE sensitivity analysis to identify service time and onboard parallelism (number of cores) as the dominant factors affecting latency and robustness. The proposal emphasizes practical benefits, enabling resource-scaling studies, generating CDFs of completion times without costly physical prototyping, and providing actionable recommendations for sizing frontend capacity and drone processing, thereby offering a compact, analytically grounded tool for architects and researchers designing resilient UAV-assisted MEC systems.

3 Architecture

The proposed architecture was designed to represent, in a structured manner, the operation of drones acting as mobile edge nodes within the MEC architecture. Its objective is to organize the flow of tasks from initial input to distributed processing, enabling the observation of bottlenecks, load balancing, and performance metrics across different operational scenarios. To this end, the architecture is organized into three interdependent layers: Admission, Frontend, and drone processing. Each layer has specific functions that, together, reflect the system's dynamic behavior and provide a framework for SPN modeling. Figure 1 presents the proposed representation.

The admission layer is responsible for receiving the system's initial requests, serving as the entry point for tasks originating from sensors, mobile devices, or users. A variety of applications can generate these requests, each with different objectives. At this stage, tasks are queued and ordered so that arrival rates and backlogs can be controlled and analyzed. This mechanism enables the simulation of overload scenarios and the assessment of their impact on overall response time, reflecting the practical need for server proximity and high computing capacity to ensure low latency and efficient resource utilization.

The FrontEnd layer acts as a load balancer and an intermediate filter. It controls the number of tasks that can be processed simultaneously, avoiding bottlenecks and overloads at critical points in the system. At this stage, requests are processed and routed to temporary queues, awaiting allocation to available drones. The FrontEnd also enables adaptive scheduling and prioritization, making the system more flexible and efficient in response to demand fluctuations. Thus, this layer acts as a central management point, ensuring greater stability and resilience.

Finally, the Drone Processing layer concentrates the mobile edge nodes responsible for executing tasks. Each drone is equipped with computing power and wireless communication capabilities, enabling it to process requests locally and return results. This approach reduces the need for continuous transmission to the cloud, saving bandwidth and enabling real-time responses. Furthermore, the processed results can be used directly by end users or integrated into decision-support systems, thereby expanding the value of decentralized processing. Thus, the proposed architecture integrates efficiency, scalability, and resilience by leveraging drones as key elements of the MEC paradigm.

4 SPN Model

In this section, we propose an SPN-based model to represent and evaluate the performance of a drone architecture acting as edge nodes. This model captures the stochastic behavior of activities from request to distributed processing onboard the drones. The components are labeled using the following terms: admission (a), queue (q), network delay (nd), frontend process (fp), transfer time (tt), frontend capacity (fc), drone processing (dp), drone capacity (dc), and processing time (pc). Descriptions of each component are provided in Table 2. Figure 2 presents the non-absorbing SPN model developed to represent the proposed architecture.

Table 2. Description of the main components of the proposed SPN model for drones as edge nodes.

Type	Components	Description
Places	Pa	Generation of new user requests
	Paq, Pfp	Admission and frontend processing queue
	Pfc	Available capacity in the frontend
	Pdp1, Pdp2, Pdp3	Processing queue in drones A, B, and C
	Pdc1, Pdc2, Pdc3	Available capacity in the drones
	Pabs	Absorbing place representing the completion of the task
Timed Transitions	Ta, Tnd	Task arrival interval and network delay
	Ttt1, Ttt2, Ttt3	Task transfer time from the Frontend to the drones
	Tpc1, Tpc2, Tpc3	Task processing time in the drones
Place Markings	Cf, Ca, Cb, Cc	Represents resource availability and system state, respectively, for the Frontend and drones A, B, and C

The model is divided into three main functional blocks: Admission, Frontend, and Drone Processing. The Admission block begins with place `Pa`, representing task generation, followed by timed transitions (`Ta`, `Tnd`) and intermediate places (`Paq`, `Pfp`) that simulate queuing and dispatching to the Frontend. The Frontend phase is represented by place `Cf`, which distributes tasks among the three servers (`Server A`, `Server B`, and `Server C`), modeled in parallel. Each server has a set of places (`Ca`, `Cb`, `Cc`) and transitions (`Tpc1`, `Tpc2`, `Tpc3`) that represent internal processing, with tokens returning to the Frontend to simulate the continuous cycle of tasks. This non-absorbing model allows observation of the system in a steady state, which is essential for continuous performance evaluation.

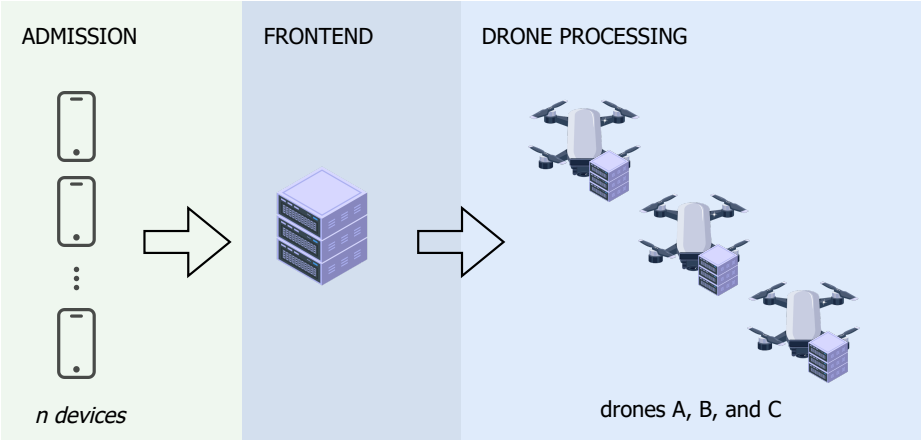


Figure 1. Proposed architecture for evaluation

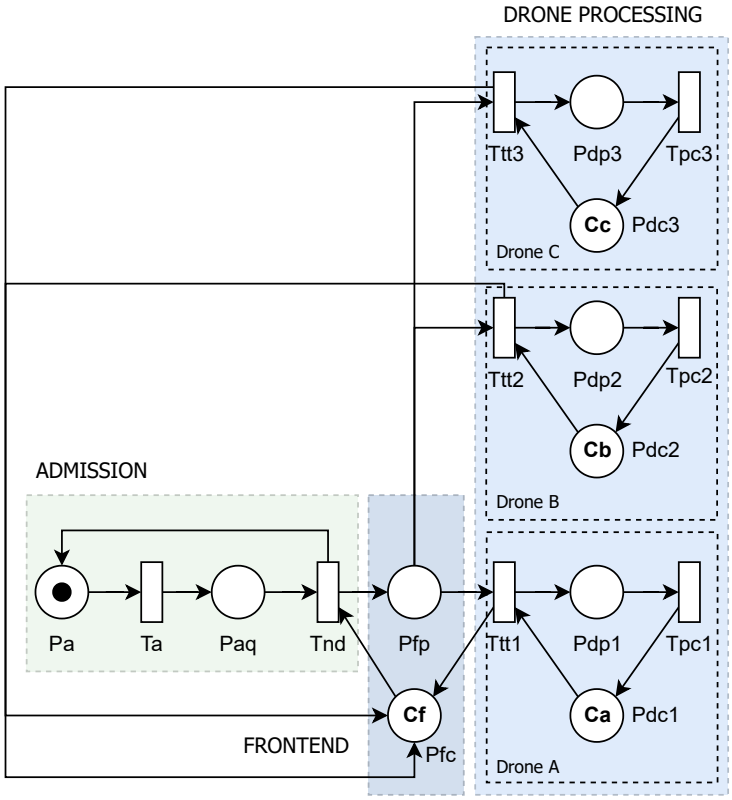


Figure 2. Non-absorbing SPN model for the proposed architecture

4.1 SPN Components

An SPN is an extension of classical Petri Nets that incorporates time and randomness to model and analyze dynamic systems with uncertain or probabilistic behaviors Guo *et al*. [2022]. The SPN combines these classical structural aspects with probability distributions associated with timed transitions, enabling a more precise quantitative and mathematical assessment of systems Balbo [2000]. Figure 3 presents the components that make up an SPN.

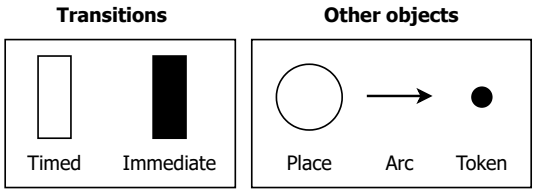


Figure 3. SPN components

The main components of an SPN include timed transitions, which represent actions or events that occur after a specified

time interval. Immediate transitions are also used, which execute instantly as soon as their conditions are met, representing decisions or events without waiting time. Places indicate the system's states or conditions. They can contain tokens, which are fundamental units of marking that represent the presence of resources, messages, or entities in those states. Arcs connect places to transitions (or vice versa) and define the direction of token flow, controlling the network dynamics. The configuration and movement of tokens along these elements describe the behavior and evolution of the modeled system Labadi *et al*. [2015]; Nguyen *et al*. [2021].

4.2 Metrics

This subsection presents the metrics to which the model was subjected. The model's quantitative evaluation was performed using the following metrics: data loss probability, throughput, MRT, and utilization. The following items describe the concept of each metric and its respective mathematical formula. In the equations, P represents a place in the network, specifically a state of the system where tokens may be present, indicating the occurrence or availability of a given condition or resource. E denotes the mathematical expectation operator, applied to the stochastic distribution of the number of tokens in a given place, allowing the estimation of expected average values for the system's behavior over time. These metrics provide a basis for analyzing the performance of the modeled architecture under different operating conditions.

The metrics considered in this study were selected because they correspond to performance indicators commonly used in SPN modeling Lopes *et al*. [2025]; Sabino *et al*. [2024]. In SPN models, these metrics are typically derived from the steady-state probabilities or reward functions associated with places and transitions, enabling researchers to analyze key aspects of system behavior, such as latency, processing capacity, reliability, and resource efficiency. When considered together, these metrics provide complementary perspectives that help characterize the overall performance of the modeled system. This approach follows common practices in SPN-based performance evaluation studies reported in the literature. Additionally, SPN modeling allows the definition of many other performance indicators depending on the model structure and the researcher's expertise in SPN-based analysis Tay [2022].

The equation 1 defines the expected number of tokens in a given place of the SPN model. In SPN-based performance evaluation, this value is obtained from the steady-state probabilities of the reachable markings of the underlying Continuous-Time Markov Chain (CTMC). Let RS denote the set of reachable markings and $\pi(m)$ the steady-state probability associated with marking m. The expected number of tokens in place P is calculated as the weighted sum of the tokens in P over all reachable markings Marsan *et al*. [1998].

$$E(P) = \sum_{m \in RS} m(P)\, \pi(m) \tag{1}$$

Equation 2 defines the loss probability (LP), which is the probability that requests are dropped due to system overload or unavailability. This metric is crucial for evaluating the system's capacity to handle high-volume data without compromising critical information. The resulting value is then converted to a percentage for easier analysis. This equation is derived from the SPN model structure, where the combination of the corresponding places in the marking indicates the system state in which requests are queued. At the same time, no frontend capacity is available, characterizing a loss condition.

$$LP = P\left((P_{\text{aq}} > 0) \wedge (P_{\text{fc}} = 0)\right) \times 100 \tag{2}$$

The equation 4 shows how throughput (TP) is calculated. Throughput represents the average number of requests successfully processed by the system per unit of time. To ensure analytical validity, the system must be in a steady state Marsan *et al*. [1998]. The effective throughput of a transition T, denoted as $X(T)$ (Equation 3), is mathematically derived from the sum of the transition's firing rates weighted by the steady-state probabilities, as follows:

$$X(T) = \sum_{m \in A(T)} \pi(m) \cdot \lambda_T(m) \tag{3}$$

Where $A(T)$ is the subset of reachable markings where transition T is enabled, and $\lambda_T(m)$ is the firing rate of transition T in marking m. In the proposed SPN model, the overall throughput corresponds to the sum of the effective throughputs of the transitions associated with the completion of processing in each drone (T_{pc1}, T_{pc2}, and T_{pc3}):

$$TP = \sum_{i=1}^{3} X(T_{pci}) \tag{4}$$

The equation 5 represents the Mean Response Time (MRT), which is the average time a task spends from the moment it is admitted until its processing is completed. This metric is important for understanding user-perceived performance, as it reflects the overall system latency. Shorter response times indicate faster task execution Little [1961].

$$MRT = \frac{\begin{array}{c} E(P_{aq}) + E(P_{fp}) + E(P_{dp1}) \\ +\ E(P_{dp2}) + E(P_{dp3}) \end{array}}{TP} \tag{5}$$

The equation 6 demonstrates the utilization metric (UN), which measures the percentage of time a processing resource is actually occupied with tasks. In this equation, Cn represents the processing capacity (e.g., number of cores) allocated to drone n. This metric helps assess load balance among drones, detect potential overloads or idleness, and guide dynamic scaling strategies. Very high utilization may indicate a risk of saturation, while very low values may suggest underutilization of available resources.

$$UN = \left(\frac{E(P_{dpn})}{Cn}\right) \times 100 \tag{6}$$

5 Case Studies

This section presents results from simulations of the proposed models, evaluating the architecture's performance with

drones as edge nodes in the MEC. Different operational scenarios were considered, varying parameters such as drone processing capacity, task service time, and the volume of requests the system accepts. The analysis was conducted using predefined quantitative metrics, enabling us to observe the system's behavior in both steady-state and finite-workload conditions. Table 3 presents the parameters used in the simulations.

The parameter values presented in this paper were defined experimentally, based on iterative simulations conducted during the model construction phase. Rather than being extracted from a specific real-world deployment, these values were chosen because they produce a representative and stable system behavior, allowing the observation of key performance phenomena such as queue buildup, resource saturation, and throughput stabilization. Small variations around these values were explored during the experimental design phase, and the results remained consistent in terms of behavioral trends, even though the absolute metric values differed. A more systematic investigation of parameter sensitivity is presented in Section 7, where the DoE analysis provides a structured evaluation of how changes in the key factors affect the Mean Response Time, further supporting the validity of the adopted baseline configuration.

Table 3. Parameters used in the model.

Type	Components	Value
Timed Transitions	Arrival Delay (AD)	0.1 ms
	Ta	0.1 ms
	Ttt1, Ttt2, Ttt3	0.01 ms
	Tpc1, Tpc2, Tpc3	1.6 ms
Place Markings	Cf	150
	Ca, Cb, Cc	4

5.1 Scenario 1 - Variation in the number of cores in the drone processing layer

This subsection analyzes the impact of varying the computational capacity of the drone processing layer, represented by the number of cores (`Cs`), on system performance. Four values of `Cs` (4.0, 6.0, 8.0, and 10.0) were considered, with the message arrival rate varying from 5 to 50 messages per second (msg/s) Bemposta Rosende *et al*. [2023]; Rey *et al*. [2025]; Shortle *et al*. [2018]. The choice of these values should allow the system's behavioral distribution to be modeled under different configurations. Table 4 presents the model parameters, including the number of components and their capacities.

Table 4. Simulation parameters for scenario 1.

Layer	Number of components	Component capacity
Frontend	1	4
Drone Processing	3	4 / 6 / 8 / 10

Figure 4(a) illustrates the utilization rate of drone processing resources as a function of the message arrival rate (msg/s), for different computing capacity configurations (`Cs` = 4, 6, 8, and 10). It can be seen that, as the arrival rate increases, drones with lower capacity (`Cs` = 4) quickly reach the 100% utilization limit, starting at 15 msg/s. Such behavior indicates early resource saturation, which compromises system scalability. In contrast, configurations with a higher number of cores (`Cs` = 10) demonstrate a greater capacity to absorb the workload, maintaining utilization at more controlled levels, even under high arrival rates. This difference underscores the importance of accurately sizing the processing layer to ensure stability and optimal performance in high-demand scenarios.

Figure 4(b) shows the system throughput, that is, the average number of messages successfully processed per second, at different arrival rates and capacity configurations. Systems with greater computational capacity (`Cs` = 10) achieve the highest throughput, exceeding 18 msg/s, while configurations with `Cs` = 4 stabilize around 10 msg/s, regardless of the increase in the input rate. Note that, for increasing arrival values, systems with `Cs` = 6 and `Cs` = 8 maintain an intermediate throughput rate, with initial growth followed by stabilization. These results demonstrate that the increase in arrival rate only translates into higher throughput when there is sufficient capacity to process the requests; otherwise, the system reaches a maximum and stops scaling.

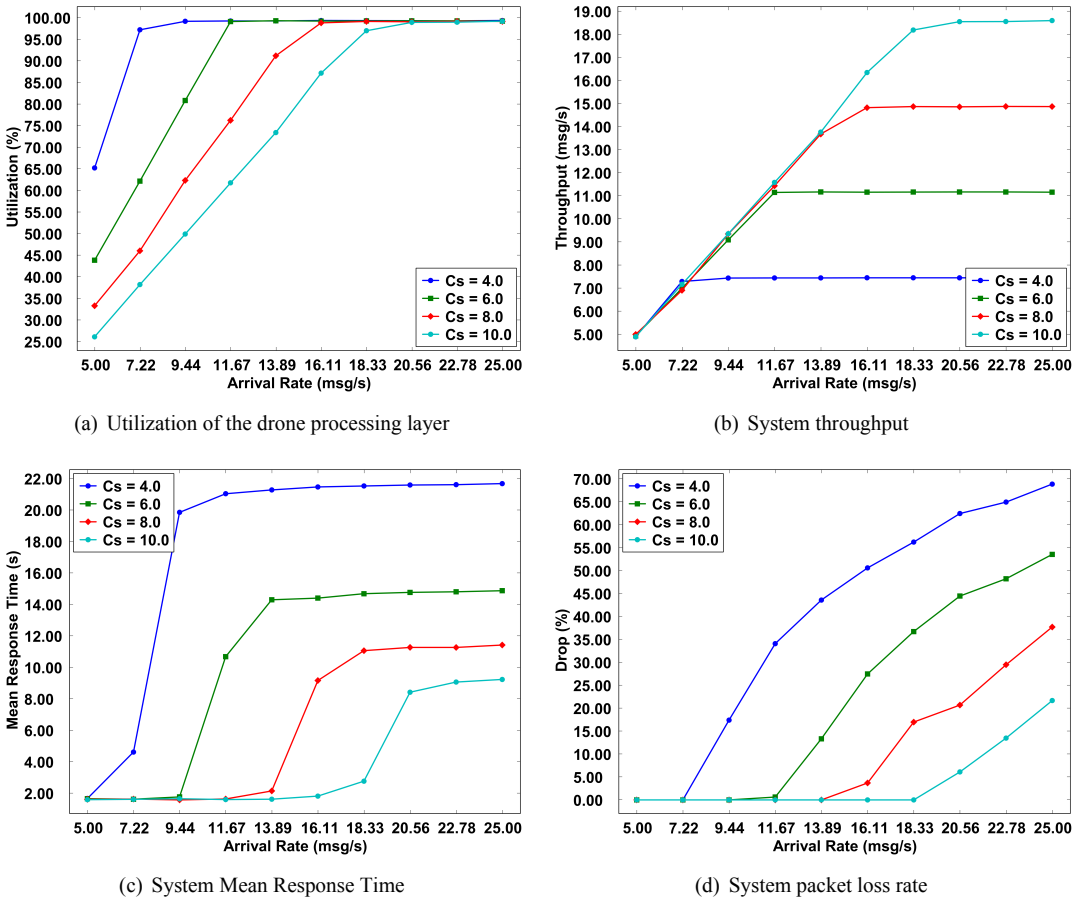
Figure 4(c) shows the MRT, which represents the interval between task admission and completion. The curve for `Cs` = 4 shows a sharp, continuous increase, exceeding 22 seconds at higher arrival rates, reflecting an overloaded, slow system. On the other hand, systems with `Cs` = 6, 8, and 10 present significantly shorter and more stable response times. In particular, `Cs` = 10 maintains the mean time below 4 seconds even under high load, demonstrating superior processing capacity. This response time behavior reinforces the need for a robust computing infrastructure to ensure agile delivery of results.

Figure 4(d) represents the percentage of packet loss in the system, indicating the proportion of messages discarded due to a lack of processing capacity. Configurations with `Cs` = 4 record losses exceeding 80% at high arrival rates, which seriously compromise system reliability. As computing capacity increases, packet loss decreases. With `Cs` = 10, the loss remains below approximately 60%, even under high load. The curves demonstrate that system scalability is directly related to its ability to avoid drops, making this metric a fundamental indicator of the architecture's robustness under various operational demands. The reduction in losses with greater capacity highlights the direct impact of adequate sizing on service quality.

5.2 Scenario 2 - Variation in service time in the drone processing layer

In this section, we investigate the impact of varying service time (TSE) in the drone processing layer on system performance. TSE represents the average time a drone takes to complete a task and is critical to the proposed system's efficiency. The analysis used Mercury simulations, with TSE varying from 0.2 ms to 1.0 ms, keeping other parameters constant Silva *et al*. [2015]. Table 5 shows the TSE values used to evaluate the previously defined metrics. This analysis is essential for optimizing operations in high-computational-demand scenarios.

Figure 5(a) illustrates the utilization rate of drone processing resources as a function of service time variations (`TSE` =



(a) Utilization of the drone processing layer

(b) System throughput

(c) System Mean Response Time

(d) System packet loss rate

Figure 4. Performance results of the non-absorbing model varying the capacity in the drone processing layer.

Table 5. Simulation parameters for scenario 2 with varying service time.

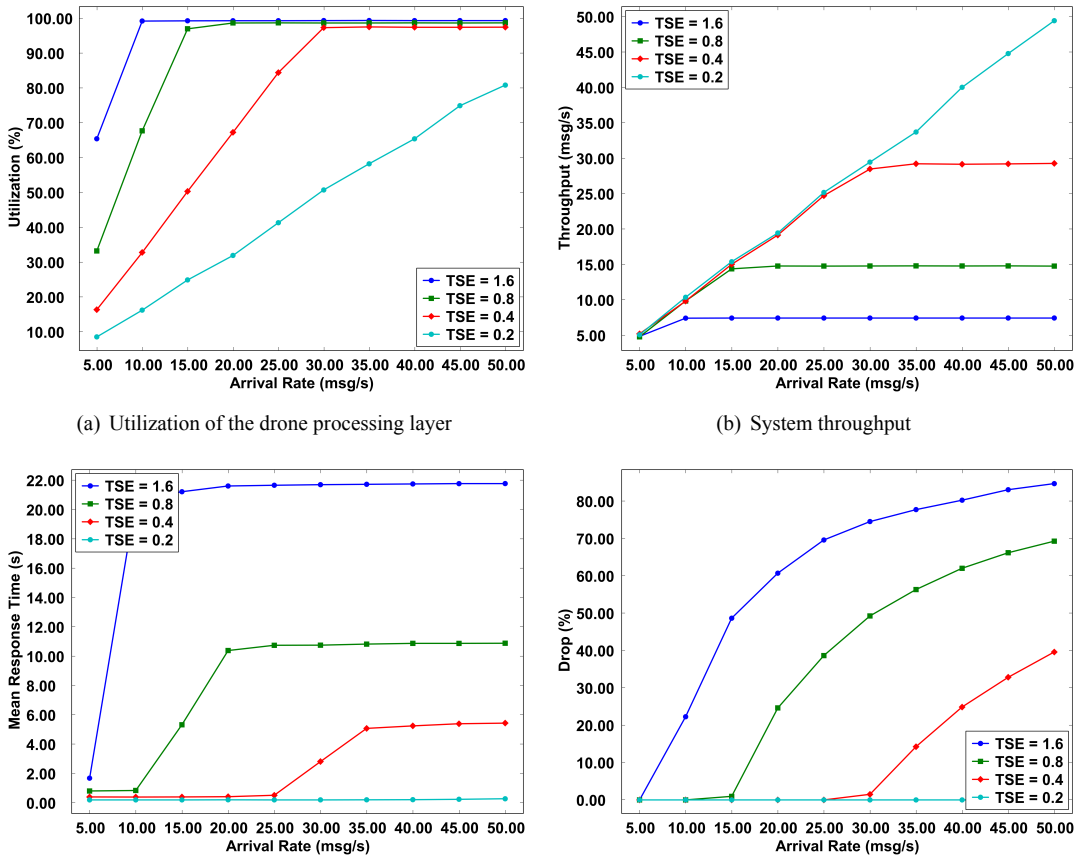
Layer	Number of queues	Service Time (ms)
Frontend	1	0.2
Drone Processing	3	0.2 / 0.4 / 0.8 / 1.6

0.2; 0.4; 0.8; 1.6 ms), for a constant arrival rate. It can be seen that as TSE increases, drone utilization quickly approaches 100%, especially at TSE = 1.6 ms. Such behavior indicates progressive resource saturation, which compromises system load balancing. In contrast, configurations with a lower TSE (0.2 ms) maintain moderate utilization, demonstrating greater availability to handle demand. This difference highlights the importance of controlling service time to ensure stability and scalability in distributed processing scenarios.

Figure 5(b) shows the system throughput, that is, the average number of messages successfully processed per second, as a function of the arrival rate, considering different service times (TSE = 0.2; 0.4; 0.8; 1.6 ms). It can be seen that, for a lower TSE (0.2 ms), the system achieves throughputs above 45 messages per second, saturating only at arrival rates above 40 messages per second, which demonstrates greater resilience in the face of increasing demands. In contrast, with TSE = 1.6 ms, the throughput stabilizes around 15 msg/s even at moderate arrival rates, revealing limitations in processing capacity that result in premature bottlenecks. These results demonstrate that reducing TSE extends the system's operational limit, maintains its efficiency under high load, and enables more effective balancing of distributed tasks.

Figure 5(c) shows the MRT, which corresponds to the interval between task admission and completion. Note that the curve for TSE = 1.6 ms exhibits a sharp, continuous increase, exceeding 20 seconds under high demand, indicating a slow, overloaded system. In contrast, systems with TSE = 0.2, 0.4, and 0.8 ms exhibit significantly shorter and more stable response times. In particular, the scenario with TSE = 0.2 ms maintains the mean time below 2 seconds, even under high loads, demonstrating superior performance and agility. This behavior reinforces the importance of optimizing service time to ensure responsiveness in edge applications.

Figure 5(d) shows the packet loss percentage in the system, indicating the proportion of messages discarded due to queue accumulation. Configurations with TSE = 1.6 ms experience losses exceeding 80% under high demand, significantly compromising the architecture's reliability. As service time decreases, packet loss decreases as well. With TSE = 0.2 ms, the loss rate remains below approximately 3% even in high-load scenarios. The curves demonstrate that the system's robustness is directly related to its ability to manage queues, making this metric a crucial indicator for assessing the quality of service under various operating conditions. The reduction in losses with optimized TSE demonstrates the direct impact on the MEC network's efficiency.



(a) Utilization of the drone processing layer

(b) System throughput

(c) Average system response time

(d) System packet loss rate

Figure 5. Performance results of the non-absorbing model varying the service time in the drone processing layer

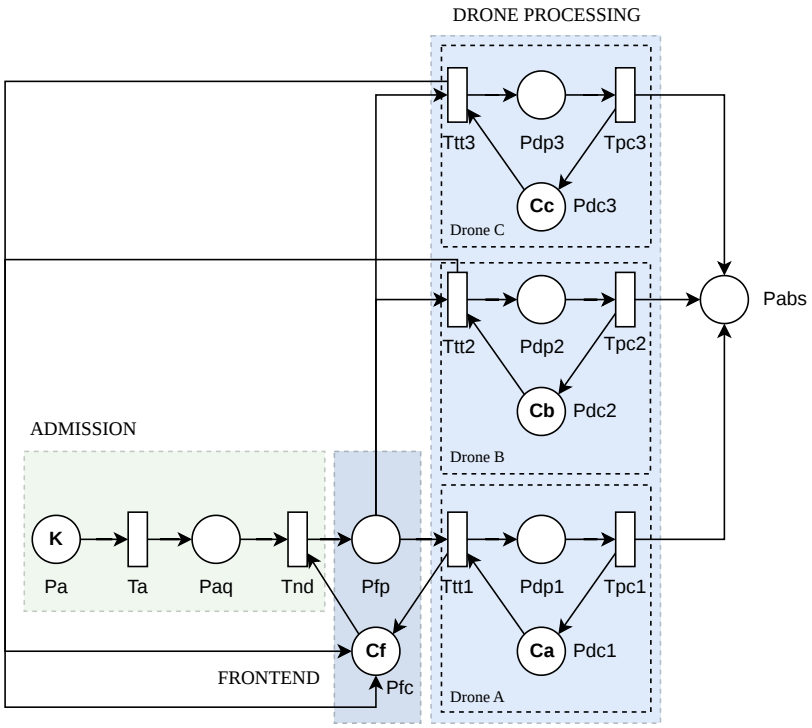


Figure 6. Absorbent SPN model

5.3 Qualitative Discussion

This subsection presents a qualitative discussion contextualizing the results obtained in the proposed model within the existing literature. Since the works in the field adopt distinct architectures, evaluation methods, and operational contexts, a direct quantitative comparison is not straightforward. Nevertheless, the behavioral trends observed in this work align with those reported in closely related studies, providing additional evidence of the consistency and validity of the proposed SPN-based framework.

Carvalho *et al*. [2020] evaluated a MEC architecture using SPN modeling and reported MRT values in the order of tens of milliseconds under moderate load, with resource utilization below 50% in well-sized configurations, a trend similarly observed in this work under low-to-moderate arrival rates with sufficient processing capacity. Sabino *et al*. [2024], who applied a comparable SPN-based framework to a drone-assisted edge computing system, reported MRT values ranging from hundreds to tens of thousands of milliseconds, depending on the number of processing cores and service time, a pattern qualitatively aligned with the results obtained here.

Additionally, Sharma and Singh [2025] evaluating a UAV-based system for forest fire detection, reported an average response time of 58.26 ms under controlled conditions with 10 UAVs. While their evaluation context differs from ours in scale and methodology, the results reinforce the relevance of response time as a critical performance indicator in UAV-assisted systems and highlight that the number of active nodes and task processing demands strongly influence latency. These qualitative comparisons demonstrate that, despite methodological differences, the results obtained in this work are consistent with trends reported in the literature, reinforcing the validity of the proposed SPN-based framework as a reliable tool for performance evaluation in UAV-assisted MEC environments.

6 Absorbent Model

The absorbent SPN model presented in this work was developed using the proposed architecture to analyze the system's transient behavior. Unlike the non-absorbent model described in Section 4, which represents continuous steady-state operation, the absorbent model incorporates an absorbent terminal location (`Pabs`), responsible for marking the completion of each task. The absence of feedback and the definition of a finite number K of tasks allows us to study the total processing time of a specific batch, generating cumulative distribution functions (CDFs) and providing detailed temporal metrics of the system. The three primary functional blocks (Admission, Frontend, and Drone Processing) are maintained to ensure consistency with the original architecture. At the same time, the absorbent location allows us to track the completion of each request individually. Figure 6 illustrates the absorbent SPN model.

The primary advantage of the absorbent model is its ability to capture transient behavior in MEC environments with drones, where variability in arrival rates and processing capacities can significantly impact performance. Each task is processed until it reaches the absorbent location, enabling the analysis of completion times, identification of bottlenecks, and accurate evaluation of quality-of-service (QoS) metrics. By removing feedback, the model avoids indefinite cycles and enables the generation of reliable temporal statistics, providing data for optimizing mobile edge architectures. Therefore, this approach is crucial for analyzing distributed processing systems under dynamic conditions, thereby supporting informed sizing and capacity planning decisions in high-demand scenarios.

6.1 Cumulative Distribution Function - CDF

Figure 7 presents the cumulative distribution function (CDF) of the time to task completion in the absorbing SPN model, considering different values of `K` (10.0, 50.0, 100.0, 150.0), which represent the total number of tasks processed by the system. The CDF indicates the probability that a task will be completed by a given point in time, providing a detailed view of the system's transient behavior. Steeper, earlier curves reflect faster completion times, while curves more shifted to the right indicate delays in task completion. This analysis is crucial for understanding system dynamics under finite loads, allowing for the evaluation of the efficiency of distributed processing on drones serving as edge nodes.

Although the values K represent increasing levels of workload in the transient analysis, their selection is not arbitrary. In performance evaluation studies, it is common practice to adopt a range of values that captures both low-load and stress conditions, allowing the observation of system behavior under progressively higher demand. In particular, moderate-to-high ranges are preferred over very small values (e.g., 1 to 5 tasks) because they better reflect realistic operating conditions and enable the emergence of queueing effects. This approach is widely adopted in the literature on queueing theory and stochastic modeling, where parameter ranges are defined to explore scalability and saturation effects around typical operating points. Feitosa *et al*. [2024]; da Silva Pinheiro *et al*. [2018]; Sabino *et al*. [2024] Therefore, the chosen interval provides a representative spectrum of system load, enabling a consistent evaluation of transient performance through cumulative distribution functions.

As `K` increases, the CDF curves shift progressively to the right, indicating that the average task completion time increases with the number of requests. This behavior is consistent with theoretical expectations, as a greater number of tasks imposes greater pressure on available processing resources, thereby increasing the time required to complete them. The results highlight the importance of adequately sizing processing capacity in MEC environments and reinforce that transient analysis using CDF provides accurate information for load planning, performance optimization, and QoS assurance in distributed drone systems.

7 DoE - Sensitivity Analysis

This section presents the system sensitivity analysis using the DoE methodology. The objective was to investigate the individual and combined effects of multiple factors on the MRT, enabling us to identify which parameters exert the greatest

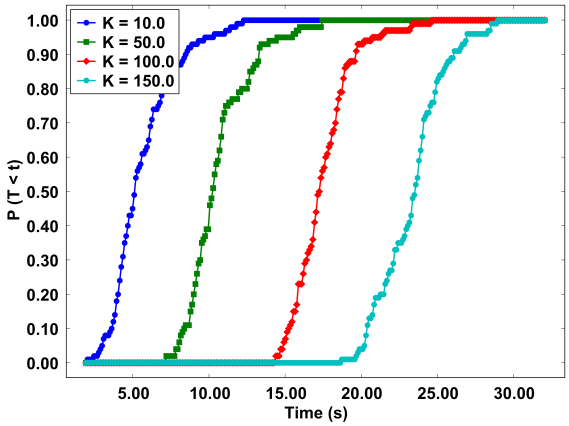


Figure 7. Cumulative distribution function for different quantities in K

influence on the proposed architecture's performance. This methodology was chosen because it allows a systematic, controlled evaluation of the variables, ensuring that both main effects and factor interactions are clearly observed. Such an approach enables a clear understanding of how different configurations influence the system's overall efficiency and guides more precise adjustments in its sizing. Furthermore, the DoE analysis serves as the primary mechanism for assessing result variability in this work, providing a structured, quantitative basis for understanding how variations in key input parameters affect system performance across a representative range of configurations.

7.1 Experiment Design

The objective of this analysis is to thoroughly investigate the main effects and interactions between factors that impact system performance. Therefore, a 2^5 complete factorial design was adopted, in which all factors are evaluated at two distinct levels (low and high), allowing the observation of not only individual effects but also second-, third-, fourth-, and even fifth-order interactions. In a 2^k factorial design, k represents the number of factors under investigation, and each factor is evaluated at two levels, resulting in 2^k experimental combinations. Thus, for $k = 5$, the design results in $2^5 = 32$ experimental configurations, enabling a comprehensive analysis of the influence of all factors and their interactions on system performance Montgomery [2017]. The use of a whole factorial arrangement enables precise quantification of each variable's contribution to overall performance. Such an approach ensures more reliable results and provides concrete support for adjusting the architecture's sizing and configuration.

The factors evaluated were: number of processing cores in the drones (`Cs`), drone service time (`TSE`), frontend service time (`TSTE`), frontend capacity (`FC`), and request arrival rate (`AD`). Each factor was tested at two levels (low and high), as shown in Table 6. The values presented in Table 6 were derived from the baseline parameters adopted in the SPN model. Specifically, the low level corresponds to 50% below the baseline, while the high level corresponds to 50% above the baseline. This configuration allows evaluating the system's sensitivity around the model's nominal operating conditions Montgomery [2017]. From this configuration, 32 simulation combinations were generated, presented in Table 7 and Table 8, allowing us to observe the system behavior under different load and configuration scenarios.

Table 6. Factors and levels considered in the simulation.

Factor	Low Level	High Level
Cs	2.0	6.0
TSE	0.8	2.4
TSTE	0.005	0.015
FC	75.0	225.0
AD	0.05	0.15

Table 7. Combinations of factor levels in the 2^5 design.

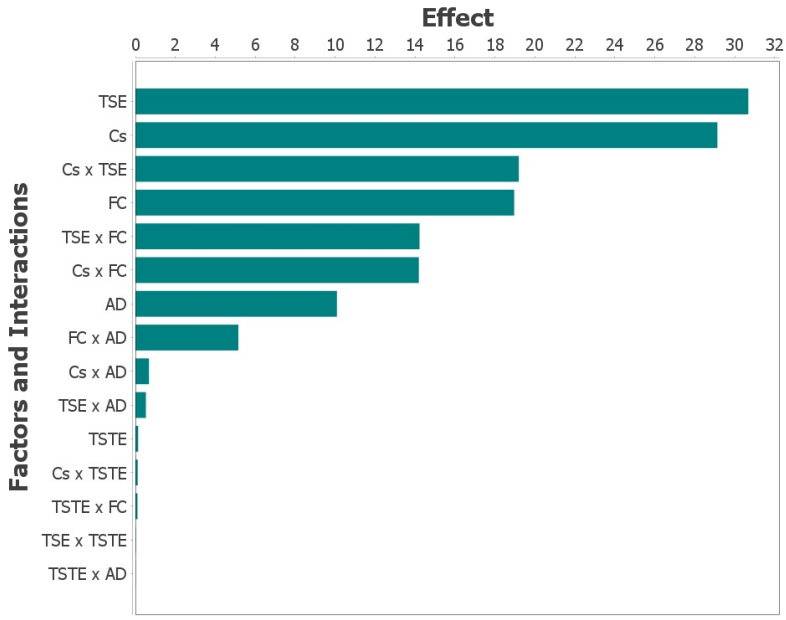
Cs	TSE	TSTE	FC	AD	MRT
2.00	0.80	0.00	75.00	0.05	10.71
2.00	0.80	0.00	75.00	0.14	1.60
2.00	0.80	0.00	225.00	0.05	29.25
6.00	2.39	0.01	225.00	0.14	2.83

Table 8. Combinations of Levels and Factors

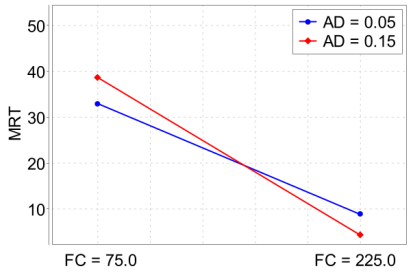
Cs	TSE	TSTE	FC	AD	MRT
2.00	0.80	0.00	75.00	0.05	10.71
2.00	0.80	0.00	75.00	0.14	1.60
2.00	0.80	0.00	225.00	0.05	29.25
2.00	0.80	0.00	225.00	0.14	1.60
2.00	0.80	0.01	75.00	0.05	10.87
2.00	0.80	0.01	75.00	0.14	1.64
2.00	0.80	0.01	225.00	0.05	30.24
2.00	0.80	0.01	225.00	0.14	1.82
2.00	2.39	0.00	75.00	0.05	32.64
2.00	2.39	0.00	75.00	0.14	31.97
2.00	2.39	0.00	225.00	0.05	91.21
2.00	2.39	0.00	225.00	0.14	86.23
2.00	2.39	0.01	75.00	0.05	32.70
2.00	2.39	0.01	75.00	0.14	31.83
2.00	2.39	0.01	225.00	0.05	91.80
2.00	2.39	0.01	225.00	0.14	87.04
6.00	0.80	0.00	75.00	0.05	1.34
6.00	0.80	0.00	75.00	0.14	0.76
6.00	0.80	0.00	225.00	0.05	0.92
6.00	0.80	0.00	225.00	0.14	0.84
6.00	0.80	0.01	75.00	0.05	0.93
6.00	0.80	0.01	75.00	0.14	0.83
6.00	0.80	0.01	225.00	0.05	1.06
6.00	0.80	0.01	225.00	0.14	0.76
6.00	2.39	0.00	75.00	0.05	12.32
6.00	2.39	0.00	75.00	0.14	2.88
6.00	2.39	0.00	225.00	0.05	30.94
6.00	2.39	0.00	225.00	0.14	3.44
6.00	2.39	0.01	75.00	0.05	12.33
6.00	2.39	0.01	75.00	0.14	3.14
6.00	2.39	0.01	225.00	0.05	30.74
6.00	2.39	0.01	225.00	0.14	2.83

7.2 Results Analysis

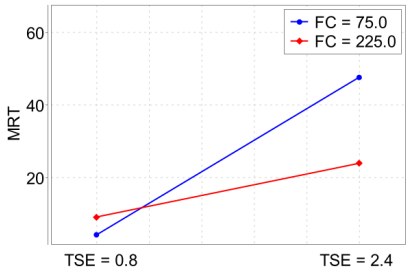
This subsection analyzes the results obtained from the factorial DoE, focusing on the impact of the evaluated factors



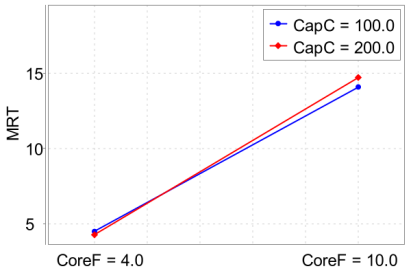
(a) Main effects on MRT.



(b) FC x AD.



(c) TSE x FC.



(d) Cs x FC.

Figure 8. Main effects and interactions between factors in relation to MRT.

on the MRT. By examining the main effects and interactions among variables, it is possible to identify the parameters that most strongly influence system performance and understand how different configurations affect overall latency in the proposed drone-assisted MEC architecture. The Figure 8 presents the results of the DoE.

The main effects plot in Figure 8(a) shows that `TSE` is the most dominant factor, with a change in `TSE` from low to high resulting in the largest increase in MRT, indicating that onboard processing delays amplify queues and total latency. In contrast, the number of cores (`Cs`) acts as a consistent mitigating factor: increasing `Cs` strongly reduces MRT, indicating that onboard parallelism is the most effective lever for reducing average latency. The `FC` and arrival rate (`AD`) factors appear to have a moderate positive effect on MRT (i.e., increasing them worsens MRT). In contrast, frontend service time (`TSTE`) has a smaller impact than the previous factors.

Figure 8(b) reveals a non-neutral interaction: the effectiveness of increasing frontend capacity depends strongly on arrival rate. When `AD` is high, increasing `FC` does not yield the expected MRT reduction in certain configurations; it may even increase MRT. This effect indicates that simple frontend capacity redistribution does not resolve the bottleneck. Figure 8(c), on the other hand, shows an adverse multiplicative effect: under high `TSE`, variations in `FC` have a much more pronounced impact on MRT, because the per-task delay in drones converts any increase in arrivals forwarded by the Frontend into increasing queues and compounded latency.

Figure 8(d) shows that scaling `Cs` (more cores in the drones) effectively mitigates frontend issues: with a high `Cs`, the MRT remains low even for high FC, while with a reduced `Cs`, the MRT increases significantly with `FC`. These results confirm that increasing embedded parallelism synergizes with frontend adjustments, making it one of the few interventions capable of making the system robust to adverse variations in `FC`/`AD`. Based on these four graphs, the practical recommendations are: prioritize reducing `TSE` and increasing `Cs` before provisioning additional `FC`; if `FC` is increased, do so only with the guarantee of proportional embedded processing capacity; consider admission policies or adaptive scaling when increasing `Cs`/`TSE` is not possible.

The amplitude of variation observed across the 32 experimental combinations further reinforces the sensitivity of the system to the evaluated factors. As shown in Tables 7 and 8, the MRT ranged from 0.76 seconds in the most favorable configuration (`Cs` = 6, `TSE` = 0.8, `FC` = 75, `AD` = 0.14) to 91.80 seconds in the most adverse one (`Cs` = 2, `TSE` = 2.39, `FC` = 225, `AD` = 0.05), representing a variation of more than two orders

of magnitude. This wide range demonstrates that parameter choices have a substantial impact on system performance, and that even moderate changes in factors such as `TSE` and `Cs` can shift the system from a highly responsive state to a severely overloaded condition.

A more detailed inspection of the results reveals additional quantitative evidence of this variability. For instance, when Cs is fixed at 2, and TSE increases from 0.8 to 2.39, the MRT jumps from 10.71 to 32.64 seconds under low arrival rate (AD = 0.05), representing an increase of approximately 205%. Similarly, when Cs is increased from 2 to 6 while keeping TSE = 2.39 and AD = 0.14, the MRT drops from 31.97 to 2.88 seconds, a reduction of over 90%. These figures confirm that the DoE analysis effectively captures the boundaries of system behavior, providing a rigorous and quantitative characterization of result variability across the full factorial space.

8 Conclusion

This work presented an SPN-based approach to model and evaluate the use of drones as nodes in Mobile Edge Computing (MEC) environments. The proposed model captured the stochastic behavior of tasks from ingestion to distributed processing across drones, accounting for processing capacity, queuing, and latency. A non-absorbing model was used for quantitative analysis of variables and an absorbing model for transient evaluation, enabling the estimation of performance metrics. The experimental results showed significant gains: in scenarios with drones equipped with more processing cores, the MRT dropped to less than 4 seconds, the loss rate decreased by more than 50%, and the throughput exceeded 18 messages per second.

Furthermore, a DoE-based sensitivity analysis showed that factors such as service time and processing capacity directly affect system efficiency. The proposed approach may be constructive for software architects, distributed systems engineers, and researchers in edge computing. Future work includes extending the model to incorporate energy consumption, drone mobility, realistic communication constraints, and scenarios with a larger number of drones to investigate system scalability further. In addition, future studies may include comparative evaluations with related approaches from the literature, as well as validation using experimental or hybrid simulation data to enhance the model's applicability to real MEC scenarios. The usefulness lies in providing an efficient overview for resource sizing, defining scaling and load-balancing strategies, and developing more resilient and efficient MEC solutions.

Acknowledgments

This work was supported by the Brazilian National Council for Scientific and Technological Development — CNPq, under grant/process number 301992/2022-3

Declarations

Funding

No funding was received to conduct this study.

Authors' Contributions

All authors contributed equally to the work.

Competing interests

The authors have no relevant financial or non-financial interests to disclose.

Availability of data and materials

Data sharing is not applicable.

References

Abubakar, A. I., Ahmad, I., Omeke, K. G., Ozturk, M., Ozturk, C., Abdel-Salam, A. M., Mollel, M. S., Abbasi, Q. H., Hussain, S., and Imran, M. A. (2023). A survey on energy optimization techniques in uav-based cellular networks: from conventional to machine learning approaches. *Drones*, 7(3):214. DOI: 10.3390/drones7030214.

Ai, Y., Liu, C., and Li, M. (2025). Energy-efficient uplink communication in uav-enabled mec networks with pinching antennas. *Drones*, 9(11):796. DOI: 10.3390/drones9110796.

Balbo, G. (2000). Introduction to stochastic petri nets. In *School organized by the European Educational Forum*, pages 84–155. Springer. DOI: 10.1007/3-540-44667-2$_3$.

Bartolini, N., Coletta, A., Maselli, G., and Piva, M. (2020). Druber: A trustable decentralized drone-based delivery system. In *Proceedings of the 6th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*, pages 1–6. DOI: 10.1145/3396864.3399706.

Bemposta Rosende, S., Ghisler, S., Fernández-Andrés, J., and Sánchez-Soriano, J. (2023). Implementation of an edge-computing vision system on reduced-board computers embedded in uavs for intelligent traffic management. *Drones*, 7(11):682. DOI: 10.3390/drones7110682.

Carvalho, D., Rodrigues, L., Endo, P. T., Kosta, S., and Silva, F. A. (2020). Mobile edge computing performance evaluation using stochastic petri nets. *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*, pages 136–143. DOI: 10.1109/ICSA-C52319.2020.00033.

da Silva Pinheiro, T. F., Silva, F. A., Fé, I., Kosta, S., and Maciel, P. (2018). Performance prediction for supporting mobile applications' offloading. *The Journal of Supercomputing*, 74(8):4060–4103. DOI: 10.1007/s11227-018-2414-6.

Emimi, M., Khaleel, M., and Alkrash, A. (2023). The current opportunities and challenges in drone technology. *Int. J. Electr. Eng. and Sustain.*, pages 74–89. DOI: 10.65998/ijees.v1i3.47.

Fedorova, A., Beliautsou, V., and Zimmermann, A. (2022). Colored petri net modelling and evaluation of drone inspection methods for distribution networks. *Sensors*, 22(9):3418. DOI: 10.3390/s22093418.

Feitosa, L., Barbosa, V., Sabino, A., Lima, L., Fé, I., Silva, B., and Silva, F. (2024). Uma comparação de múltiplas políticas de migração de contêineres suportadas pela ferramenta criu. In *Anais do XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 742–755, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbrc.2024.1467.

Guo, Q., Yu, W., Hao, F., Zhou, Y., and Liu, Y. (2022). Modelling and analysis of adaptive cruise control system based on synchronization theory of petri nets. *Electronics*, 11(21):3632. DOI: 10.3390/electronics11213632.

Hossain, R. (2022). A short review of the drone technology. *International Journal of Mechatronics and Manufacturing Technology*, 7(2):53–68. Available at:`https://www.researchgate.net/publication/362908663_A_Short_Review_of_the_Drone_Technology`.

Jeong, S., Simeone, O., and Kang, J. (2017). Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning. *IEEE Transactions on Vehicular Technology*, 67(3):2049–2063. DOI: 10.1109/tvt.2017.2706308.

Labadi, K., Benarbia, T., Barbot, J.-P., Hamaci, S., and Omari, A. (2015). Stochastic petri net modeling, simulation and analysis of public bicycle sharing systems. *IEEE Transactions on Automation Science and Engineering*, 12(4):1380–1395. DOI: 10.1109/TASE.2014.2336874.

Little, J. D. (1961). A proof for the queuing formula: L= λ w. *Operations research*, 9(3):383–387. DOI: 10.1287/opre.9.3.383.

Liu, W., Zhang, S., and Ansari, N. (2022). Secure mobile edge computing via a drone-enabled fso-based heterogeneous network. *IEEE Transactions on Vehicular Technology*, 71(11):12264–12276. DOI: 10.1109/TVT.2022.3194407.

Liu, Y., Li, H., Vasilakos, X., Hussain, R., and Simeonidou, D. (2025). Cooperative task offloading through asynchronous deep reinforcement learning in mobile edge computing for future networks. In *ICC 2025-IEEE International Conference on Communications*, pages 1390–1395. IEEE. DOI: 10.1109/icc52391.2025.11160823.

Lopes, L. S., Araújo, J. M., Lima, L. N., Barbosa, V., Sabino, A., Rocha Filho, G. P., and Silva, F. A. (2025). Performance evaluation of a camera surveillance system in smart buildings using queuing models. *Journal of Internet Services and Applications*, 16(1):480–491. DOI: 10.5753/jisa.2025.5076.

Luo, S., Li, H., Wen, Z., Qian, B., Morgan, G., Longo, A., Rana, O., and Ranjan, R. (2021). Blockchain-based task offloading in drone-aided mobile edge computing. *IEEE Network*, 35(1):124–130. DOI: 10.1109/MNET.011.2000222.

Marsan, M. A., Balbo, G., Conte, G., Donatelli, S., and Franceschinis, G. (1998). Modelling with generalized stochastic petri nets. *ACM SIGMETRICS performance evaluation review*, 26(2):2. DOI: 10.1145/288197.581193.

Miao, Y., Hwang, K., Wu, D., Hao, Y., and Chen, M. (2023). Drone swarm path planning for mobile edge computing in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 19(5):6836–6846. DOI: 10.1109/TII.2022.3196392.

Montgomery, D. C. (2017). *Design and analysis of experiments*. John wiley & sons. DOI: 10.2307/1269246.

Nguyen, T. A., Fe, I., Brito, C., Kaliappan, V. K., Choi, E., Min, D., Lee, J. W., and Silva, F. A. (2021). Performability evaluation of load balancing and fail-over strategies for medical information systems with edge/fog computing using stochastic reward nets. *Sensors*, 21(18):6253. DOI: 10.3390/s21186253.

Park, S., Kim, H. T., Lee, S., Joo, H., and Kim, H. (2021). Survey on anti-drone systems: Components, designs, and challenges. *IEEE access*, 9:42635–42659. DOI: 10.1109/access.2021.3065926.

Rey, L., Bernardos, A. M., Dobrzycki, A. D., Carramiñana, D., Bergesio, L., Besada, J. A., and Casar, J. R. (2025). A performance analysis of you only look once models for deployment on constrained computational edge devices in drone applications. *Electronics*, 14(3):638. DOI: 10.3390/electronics14030638.

Rojas-Perez, L. O. and Martínez-Carranza, J. (2021). On-board processing for autonomous drone racing: An overview. *Integration*, 80:46–59. DOI: 10.1016/j.vlsi.2021.04.007.

Sabino, A., Lima, L. N., Brito, C., Feitosa, L., Caetano, M. F., Barreto, P. S., and Silva, F. A. (2024). Forest fire monitoring system supported by unmanned aerial vehicles and edge computing: a performance evaluation using petri nets. *Cluster Computing*, 27(7):9735–9755. DOI: 10.1007/s10586-024-04504-5.

Sharma, A. and Singh, P. K. (2025). Uav-based framework for effective data analysis of forest fire detection using 5g networks: An effective approach towards smart cities solutions. *International Journal of Communication Systems*, 38(1):e4826. DOI: 10.1002/dac.4826.

Shnaiwer, Y. N., Kouzayha, N., Masood, M., Kaneko, M., and Al-Naffouri, T. Y. (2022). Multihop task routing in uav-assisted mobile-edge computing iot networks with intelligent reflective surfaces. *IEEE Internet of Things Journal*, 10(8):7174–7188. DOI: 10.1109/jiot.2022.3228863.

Shortle, J. F., Thompson, J. M., Gross, D., and Harris, C. M. (2018). *Fundamentals of queueing theory*. John Wiley & Sons. DOI: 10.2307/2344588.

Silva, B., Matos, R., Callou, G., Figueiredo, J., Oliveira, D., Ferreira, J., Dantas, J., Lobo, A., Alves, V., and Maciel, P. (2015). Mercury: An integrated environment for performance and dependability evaluation of general systems. In *Proceedings of industrial track at 45th dependable systems and networks conference, DSN*, pages 1–4. DOI: 10.13140/rg.2.1.1369.8325.

Statista (2023). Drone market revenue worldwide from 2018 to 2029. Available at:`https://www.statista.com/forecasts/1399063/drone-market-revenue-worldwide`. Accessed: 2025-07-11.

Sun, Z., Guan, X., Lin, Z., Fang, Z., Cai, X., Chen, Z., Liu, F., Cui, H., Xiong, J., Ni, W., *et al*. (2025). Intra-dp: A high performance collaborative inference system for mobile

edge computing. *arXiv preprint arXiv:2507.05829*. DOI: 10.48550/arxiv.2507.05829.

Tay, Y. C. (2022). *Analytical performance modeling for computer systems*. Springer Nature. DOI: 10.1007/978-3-031-01795-7.

Urvashi, n. and Bansal, S. (2025). Performance optimization and design of a fire extinguisher wireless sensor drone system using petri nets modeling and pso algorithm. *IEEE Sensors Journal*, 25(3):4801–4810. DOI: 10.1109/JSEN.2024.3513447.

Zeng, Y. and Tang, J. (2023). Mec-assisted real-time data acquisition and processing for uav with general missions. *IEEE Transactions on Vehicular Technology*, 72(1):1058–1072. DOI: 10.1109/TVT.2022.3203704.

Zhang, J., Zhou, L., Tang, Q., Ngai, E. C., Hu, X., Zhao, H., and Wei, J. (2019). Stochastic computation offloading and trajectory scheduling for uav-assisted mobile edge computing. *IEEE Internet of Things Journal*, 6(2):3688–3699. DOI: 10.1109/JIOT.2018.2890133.