Universidade Federal do Piauí
Centro de Ciências da Natureza
Programa de Pós-Graduação em Ciência da Computação

Availability and Performance Evaluation of VANETs Based on Stochastic Petri Nets

Luis Guilherme Sousa e Silva

Número de Ordem PPGCC: M001

Picos-PI, Dezembro de 2025

Luis Guilherme Sousa e Silva

Availability and Performance Evaluation of VANETs Based on Stochastic Petri Nets

Defesa de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da UFPI (área de concentração: Sistemas de Computação), como parte dos requisitos necessários para a obtenção do Título de Mestre em Ciência da Computação.

Universidade Federal do Piauí – UFPI

Centro de Ciências da Natureza

Programa de Pós-Graduação em Ciência da Computação

Orientador: Prof. Dr. Francisco Airton Pereira da Silva

Picos-PI

Dezembro de 2025

FICHA CATALOGRÁFICA
Universidade Federal do Piauí
Sistema de Bibliotecas UFPI - SIBi/UFPI
Biblioteca Setorial do CCN

S586a Silva, Luís Guilherme Sousa e.
Availability and performance evaluation of VANET'S based on Stochastic Petri Nets / Luís Guilherme Sousa e Silva. -- 2025.
128 f.: il.

Dissertação (Mestrado) - Universidade Federal do Piauí. Centro de Ciências da Natureza. Programa de Pós-Graduação em Ciências da Computação, Picos, 2025.
"Orientador: Prof. Dr. Francisco Airton Pereira da Silva".

1. Rede Veicular. 2. Computação em borda. 3. Redes de Petri Estocásticas - SPN's. I. Silva, Francisco Airton Pereira da. II. Titulo.

CDD 005.43

Bibliotecária: Caryne Maria da Silva Gomes - CRB3/1461

Luis Guilherme Sousa e Silva

Availability and Performance Evaluation of VANETs Based on Stochastic Petri Nets

Defesa de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da UFPI (área de concentração: Sistemas de Computação), como parte dos requisitos necessários para a obtenção do Título de Mestre em Ciência da Computação.

Documento assinado digitalmente
gov.br
FRANCISCO AIRTON PEREIRA DA SILVA
Data: 10/12/2025 16:43:24-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Francisco Airton Pereira da Silva
Orientador

Documento assinado digitalmente
gov.br
JULIANA OLIVEIRA DE CARVALHO
Data: 11/12/2025 10:05:23-0300
Verifique em https://validar.iti.gov.br

Profa. Dra. Juliana Oliveira de Carvalho
Membro da Banca

Documento assinado digitalmente
gov.br
RAYNER GOMES SOUSA
Data: 08/01/2026 11:57:55-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Rayner Gomes Sousa
Membro da Banca

Documento assinado digitalmente
gov.br
CHRISTIAN RODOLFO ESTEVE ROTHENBERG
Data: 21/01/2026 12:40:00-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Christian Rodolfo Esteve Rothenberg
Membro da Banca

Picos-PI
Dezembro de 2025

Agradecimentos

Agradeço primeiramente a Deus por ter me dado saúde, por me conceder força, sabedoria e serenidade em todos os momentos desta jornada. Sua presença foi essencial para que eu mantivesse a fé e a determinação diante dos desafios encontrados ao longo do caminho. Aos meus pais, José Luis e Maria de Jesus, por todo amor, força, apoio incondicional e ensinamentos que servem de base para cada conquista da minha vida. À minha irmã Gêysa Janne, ao meu cunhado Levi França e aos demais familiares, pela compreensão, carinho e incentivo constantes. À minha namorada, Thais Alves, pela paciência, companheirismo e apoio emocional durante esse tempo. Sua presença trouxe equilíbrio, conforto e motivação nos momentos mais desafiadores desta caminhada. Aos meus amigos de caminhada, em especial Gabriel Holanda, que foi o principal incentivador para que eu ingressasse no mestrado, e também Israel Cardoso, Lucas Vinícius e Carlos Vitor, pela amizade, pelas conversas inspiradoras e pelo companheirismo em cada etapa dessa trajetória acadêmica. Ao meu orientador, Dr. Francisco Airton, pelos ensinamentos, orientações, dedicação e confiança depositada em meu trabalho. Agradeço também aos demais professores, pelo tempo pelas contribuições valiosas e por todo conhecimentos compartilhado ao longo do curso. Agradeço a CAPES e FAPEPI pelo apoio financeiro para realização deste trabalho de pesquisa. Por fim, agradeço a todos que, de alguma forma, estiveram comigo durante essa importante fase da minha vida. Este trabalho é fruto não apenas de esforço individual, mas da soma de afetos, inspirações e aprendizados que recebi de pessoas incríveis ao longo do caminho. A todos, meu sincero e eterno muito obrigado.

A dúvida é o caminho natural para a conhecimento
(Autor Desconhecido)

Resumo

A mobilidade urbana atual exige requisitos de continuidade de serviço, adaptabilidade, resiliência e tolerância a falhas em sistemas computacionais distribuídos. Nesse cenário, *Vehicular Ad Hoc Networks* (VANETs), integradas à computação em borda e organizadas segundo arquiteturas de microsserviços, sustentam aplicações de monitoramento e controle de tráfego sob condições operacionais variáveis. A limitação de recursos na borda inviabiliza o uso de estratégias tradicionais de redundância baseadas em replicação física, enquanto a variabilidade da carga nas RSUs demanda mecanismos de elasticidade capazes de ajustar a capacidade de processamento ao longo do tempo. Diante desta complexidades, torna-se necessário empregar abordagens analíticas que permitam avaliar, de forma preditiva, a confiabilidade e o desempenho do sistema. Nesse contexto, esta dissertação desenvolve modelos baseados em Redes de Petri Estocásticas (SPNs) para representar e analisar a disponibilidade e o desempenho de uma arquitetura VANET composta por múltiplas *Road Side Units* (RSUs) conectadas a um servidor de borda. O modelo de disponibilidade contempla falhas físicas, lógicas e de comunicação, avaliando a continuidade operacional por meio da migração de máquinas virtuais entre RSUs. A modelagem emprega parâmetros de tempo médio entre falhas e de reparo, e os resultados mostram que a migração mantém a disponibilidade e reduz o tempo de recuperação mesmo sob falhas simultâneas. Para a avaliação de desempenho, foi desenvolvido um modelo SPN que representa a adaptação da capacidade de processamento nas RSUs por meio de autoescalonamento horizontal, que ajusta automaticamente a alocação de instâncias conforme a variação da carga de trabalho, com reinstanciação de máquinas virtuais. Os cenários analisam variações de carga e limites de escalonamento, evidenciando condições de saturação quando esses limites são mal configurados e melhor equilíbrio operacional com os mecanismos de autoescalonamento. A análise de sensibilidade, conduzida por meio de *Design of Experiments* (DoE), mostrou que variações em parâmetros como número inicial de máquinas virtuais e tamanho de fila afetam diretamente as métricas de resposta e descarte, permitindo orientar a configuração de políticas operacionais. A análise de sustentabilidade estende o modelo de desempenho com métricas energéticas e ambientais, relacionando consumo de energia e emissões de carbono ao comportamento do sistema. A calibração com dados experimentais do *Pasid-Validator* confirma a aderência do modelo analítico, e os resultados indicam que políticas dinâmicas de autoescalonamento com reinstanciação reduzem o consumo energético e a pegada de carbono sem comprometer a estabilidade operacional.

Palavras-chaves: Redes Veiculares Ad Hoc; Computação em Borda; Microsserviços; Redes de Petri Estocásticas; Disponibilidade; Desempenho; Sustentabilidade; Migração de VMs; Autoescalonamento; Reinstanciação; Análise de Sensibilidade; Eficiência Energética; Validação.

Abstract

Current urban mobility demands continuity of service, adaptability, resilience, and fault tolerance in distributed computing systems. In this scenario, Vehicular Ad Hoc Networks (VANETs), integrated with edge computing and organized according to microservice architectures, support traffic monitoring and control applications under variable operational conditions. The limitation of resources at the edge makes the use of traditional redundancy strategies based on physical replication unfeasible, while the variability of the load on the Road Side Units (RSUs) demands elasticity mechanisms capable of adjusting processing capacity over time. Given these complexities, it becomes necessary to employ analytical approaches that allow for the predictive evaluation of system reliability and performance. In this context, this dissertation develops models based on Stochastic Petri Nets (SPNs) to represent and analyze the availability and performance of a VANET architecture composed of multiple Road Side Units (RSUs) connected to an edge server. The availability model considers physical, logical, and communication failures, evaluating operational continuity through the migration of virtual machines between RSUs. The modeling employs mean time between failures and time to repair parameters, and the results show that migration maintains availability and reduces recovery time even under simultaneous failures. For performance evaluation, an SPN model was developed that represents the adaptation of processing capacity in the RSUs through horizontal autoscheduling, which automatically adjusts instance allocation according to workload variations, with virtual machine re-instantiation. The scenarios analyze load variations and scaling limits, highlighting saturation conditions when these limits are poorly configured and better operational balance with autoscheduling mechanisms. Sensitivity analysis, conducted through Design of Experiments (DoE), showed that variations in parameters such as the initial number of virtual machines and queue size directly affect response and discard metrics, allowing for the guidance of operational policy configuration. Sustainability analysis extends the performance model with energy and environmental metrics, relating energy consumption and carbon emissions to system behavior. Calibration with experimental data from the PASID-VALIDATOR confirms the adherence of the analytical model, and the results indicate that dynamic autoscaling policies with reinstantiation reduce energy consumption and carbon footprint without compromising operational stability.

Keywords: Vehicular Ad Hoc Networks; Edge Computing; Microservices; Stochastic Petri Nets; Availability; Performance; Sustainability; VM Migration; Autoscaling; Reinstantiation; Sensitivity Analysis; Energy Efficiency; Validation.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

DoE	Design of Experiments
DP	Discard Probability
DSRC	Dedicated Short Range Communications
GSPN	Generalized Stochastic Petri Nets
IoT	Internet of Things
ITS	Intelligent Transportation System
MRT	Mean Response Time
MTTF	Mean Time to Failure
MTTR	Mean Time to Repair
NVMU	Number of Virtual Machines in Use
OBU	On-Board Unit
PN	Petri Net
QoS	Quality of Service
RSU	Road Side Units
RTA	Regional Trusted Authority
SPN	Stochastic Petri Net
TA	Trusted Authority
TP	Throughput
U	Utilization
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
VANET	Vehicular Ad Hoc Network
VM	Virtual Machines
WAVE	Wireless Access in Vehicular Environments

Sumário

1 Introduction

The expansion of connected urban mobility and the growing integration between vehicles and road infrastructure have increased the demand for distributed systems capable of operating with low latency, adaptability, and service continuity (HASROUNY et al., 2017). VANETs are designed for communication between vehicles and road infrastructure, with each vehicle acting as a mobile node capable of transmitting. The main objective of VANETs is to improve traffic safety, efficiency, and comfort by enabling the exchange of information about accidents, congestion, weather conditions, alternative routes, and other critical events (SINGH et al., 2022).

RSUs play a key role in this ecosystem, acting as access points that extend connectivity and facilitate the exchange of information between vehicles and traffic management systems. This integration enables real-time sharing of data on road conditions, safety, and urban mobility (NI; ZHAO; CAI, 2021). VANETs have been used as the basis for implementing decentralized traffic monitoring and control solutions, supported by RSUs interconnected to edge servers. These units process locally generated data, reducing the dependence on communication with central servers (CHOUDHARY; DORLE, 2021). In environments subject to continuous traffic variations, physical failures and intermittent connectivity, ensuring the operational stability of highway network architectures requires mechanisms that respond to unpredictable events and adapt computing resources dynamically (Ferreira, W., 2023).

Architectures based on VANETs operate under conditions of high variability, in which failure events and demand peaks occur in a non-deterministic manner. RSUs are subject to logical degradation (in software and applications), physical failures (in hardware) and connectivity interruptions, while the processing load can fluctuate abruptly depending on the flow of vehicles (APM, 2022). The absence of automatic reconfiguration mechanisms in these scenarios can compromise the continuity of distributed services, leading to performance degradation. Furthermore, the cost of implementing practical evaluation of these solutions in real urban environments limits the ability to test multiple configurations or anticipate the impact of critical events (VERMA, 2023). Given these constraints, formal models that represent the architecture's behavior under different operational conditions and provide assistance for the system's technical planning are necessary.

The adoption of virtualization in RSUs allows the execution of isolated microservices, allocated in dynamically managed virtual machines (MANIVANNAN; MONI; ZEADALLY, 2020). This approach enables the implementation of two complementary mechanisms: The migration of VMs between RSUs in the event of failures and horizontal autoscaling with

reinstantiation to adapt to traffic load. Inter-RSU migration seeks to ensure the continuity of services in the event of physical or logical failures, while inter-RSU autoscaling responds to variations in computational demand caused by fluctuations in vehicular flow (ANWER; GUY, 2014).

The effectiveness of these mechanisms depends on the system's ability to detect failures, monitor resource utilization, and allocate or reallocate VMs within time constraints that preserve service continuity. However, evaluating such strategies in real-world scenarios is highly complex. For instance, reproducing a scenario in which an RSU experiences simultaneous hardware degradation and connectivity loss, while also facing a surge in vehicle-generated data, would require specialized infrastructure, synchronized fault injection, and controlled network conditions. These requirements translate into high financial costs, logistical challenges, and limited reproducibility, especially in dense urban environments with unpredictable traffic patterns. This corroborates the use of analytical modeling through SPNs, which allow the representation of concurrent behaviors, timed events, and stochastic dynamics in distributed environments (ARAÚJO et al., 2021a).

1.1 Problem

The operational stability of ad hoc vehicular networks based on edge units is directly linked to the availability of the RSUs that build the system's fixed infrastructure (HUSSAIN; HUSSAIN; ZEADALLY, 2019). However, these units are subject to different failure modes, including physical failures associated with the hardware, logical failures resulting from degradation of the operating system or services being executed, and communication failures caused by loss of connectivity with the adjacent network (SHEN et al., 2019). The unavailability of an RSU compromises the execution of the microservices allocated to it, interrupts real-time data processing, and can directly affect the continuity of service on road sections under its responsibility (RAUT; DEVANE, 2017). Given these architectures' decentralized nature, the failure of a single node can interrupt local data flows, unbalance the processing load on other units, and compromise the network's functional coverage.

The absence of autonomous mechanisms for reallocating services in response to this type of event limits the system's ability to maintain its minimum functionality under adverse conditions (HUSSAIN; HUSSAIN; ZEADALLY, 2019). Solutions based on physical replication, such as cold, warm, or hot standby configurations, have been used to mitigate node unavailability in distributed systems (FEITOSA et al., 2021). However, such strategies imply duplication of resources and simultaneous maintenance of idle instances, resulting in a significant increase in operational and acquisition costs. In vehicular environments, where infrastructure is limited and available computing resources at the edge are restricted, the adoption of these approaches becomes unfeasible on a large scale (Blessy A; S, 2023).

In addition, these methods assume a static and pre-defined allocation of redundancy, with no capacity for dynamic reconfiguration in response to unforeseen failures. The lack of operational flexibility limits their applicability in scenarios of topological instability, such as those found in VANETs deployed in heterogeneous urban areas (QIONG et al., 2023).

The computational load imposed on RSUs in urban environments varies significantly over time, reflecting the vehicular mobility patterns characteristic of densely occupied regions. During peak periods, such as peak hours in the early morning and late afternoon, the volume of requests may exceed the previously allocated processing capacity, causing queues to build up and increasing response times (ZHANG et al., 2016). On the other hand, during periods of low demand, maintaining idle active instances compromises the efficiency of using available resources. The absence of automatic scaling and conditional reallocation mechanisms compromises the elasticity of the system, reduces its adaptability, and generates risks of saturation or underutilization, the direct effects of which are manifested in performance degradation and an increase in the rate of request discards (OUHMIDOU et al., 2023).

Despite recent advances, the solutions proposed in the literature often treat the aspects of availability and performance in isolation, disregarding the interdependence between recovery mechanisms through migration and capacity adaptation through scaling (PARASHAR; TIWARI, 2023). In addition, most of the work assumes static environments or ideal configurations, neglecting the simultaneous occurrence of failures, resource saturation, and topological instabilities typical of vehicular scenarios (KARABULUT et al., 2023). The experimental evaluation of these architectures in real urban traffic conditions requires specialized infrastructure, control over external variables, and replicability of critical events, which results in high costs and logistical limitations (FÉ et al., 2022a). In traffic monitoring systems, failures that are not correctly handled can compromise incident detection, intelligent traffic light control, and response to emergency events, increasing the risks to the physical integrity of road users (ARAÚJO et al., 2021a).

In this context, analytical models based on SPNs offer a structured and formal alternative to represent systems with concurrent behaviors, dynamic reconfiguration, timed failures, and stochastic adaptation. Such models allow the analysis of complex scenarios, provide quantitative metrics of reliability and performance, and enable the application of sensitivity strategies such as DoE, without relying on physical implementation or large-scale empirical simulations (NARESH et al., 2024). The literature reviewed often treats availability and performance in isolation, disregarding the critical interdependence between migration and scaling. Traditional strategies based on physical replication impose static duplication of resources, an approach that is unfeasible in edge infrastructures and with devices that have limited capacity. The central problem lies in the lack of mechanisms that can maintain the minimum functionality of the system under the simultaneous

occurrence of failures, resource saturation, and topological instability. Migration of VMs in RSUs is essential to ensure the immediate continuity of microservices after physical or logical failures. This study sought to address this gap by employing SPNs to model and quantify the integrated behavior of elasticity and resilience in dynamic vehicular environments. Therefore, the main question of this research is: **How can we evaluate performance, availability, and sustainability metrics in VANET systems without prior physical infrastructure?**

1.2 Objetives

This work has as its main objective to support administrators of computing infrastructures in the planning and evaluation of VANET systems based on RSUs, with emphasis on the analysis of availability and computational performance of the proposed architectures.

The specific objectives of this work are:

- Evaluate the impact of virtual machine migration mechanisms on service continuity in vehicular architectures composed of multiple RSUs, modeling different failure modes (physical and logical), recovery rates, and topological scenarios, using SPN;
  - A paper was written based on this specific objective and published in Sensors (SILVA et al., 2023).
- Analyze the dynamic performance behavior of RSU units subjected to variations in the computational load generated by vehicular traffic, considering horizontal scalability limits, processing times, and conditional reinstantiation policies, aiming to estimate performance metrics such as response time, discard rate, and resource utilization;
  - A paper was written based on this specific objective and published in the XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SILVA et al., 2024).
- Conduct a computational sustainability and performance analysis, estimating energy consumption and the balance between operational efficiency and performance based on the adopted scaling and reinstantiation strategies, and validate the model in operating scenarios under different levels of load, verifying its adherence to expected behaviors.
  - A paper was written based on this specific objective and published in International Journal of Communication Systems (SILVA et al., 2025).

1.3 Out of Scope

Some topics are outside the scope of this work and, therefore, have not been addressed in this work to keep it objective. Some of these topics are listed below:

- Machine learning techniques: The work does not address traffic prediction, event classification, or supervised or unsupervised techniques for dynamic parameter adjustment. The focus of this dissertation is on formal analytical modeling with SPNs, without any component of statistical inference based on historical data.

- Security: Aspects related to confidentiality, integrity, authentication, encryption, or intrusion detection were not considered. Security modeling in distributed vehicular systems requires representation of highly random and adversarial events, which requires another type of formal approach. SPN models applied in this work do not contemplate attacks or defense strategies.

- Data storage: The data stream generated by vehicles and processed by RSUs is not persisted or forwarded to external repositories within the scope of this study. Metrics related to write latency, replication, or discard policies were not considered. The proposal focuses solely on the processing and response of the system under load and failure.

- Vehicle communication or mobility protocols: The work does not model or evaluate specific routing, media access, or handover protocols between RSUs. The communication layer is considered functional and stable in the model, and the focus is on analyzing the computational behavior of the architecture under failures and load variations.

- Vehicle mobility: The work does not represent the physical displacement of vehicles, nor does it model urban mobility patterns, average speed, routes, or coverage zone transitions between RSUs. Vehicles are considered only as sources of requests, with arrival rates varying over time. The spatial dynamics of traffic are not part of the analytical model, being abstracted through parameterized distributions.

1.4 Methodology

The main objective of this work is to develop analytical models to evaluate the availability and performance of a distributed vehicular architecture based on RSUs and edge computing. Figure 1 presents a flowchart that summarizes the methodological approach adopted in this work.

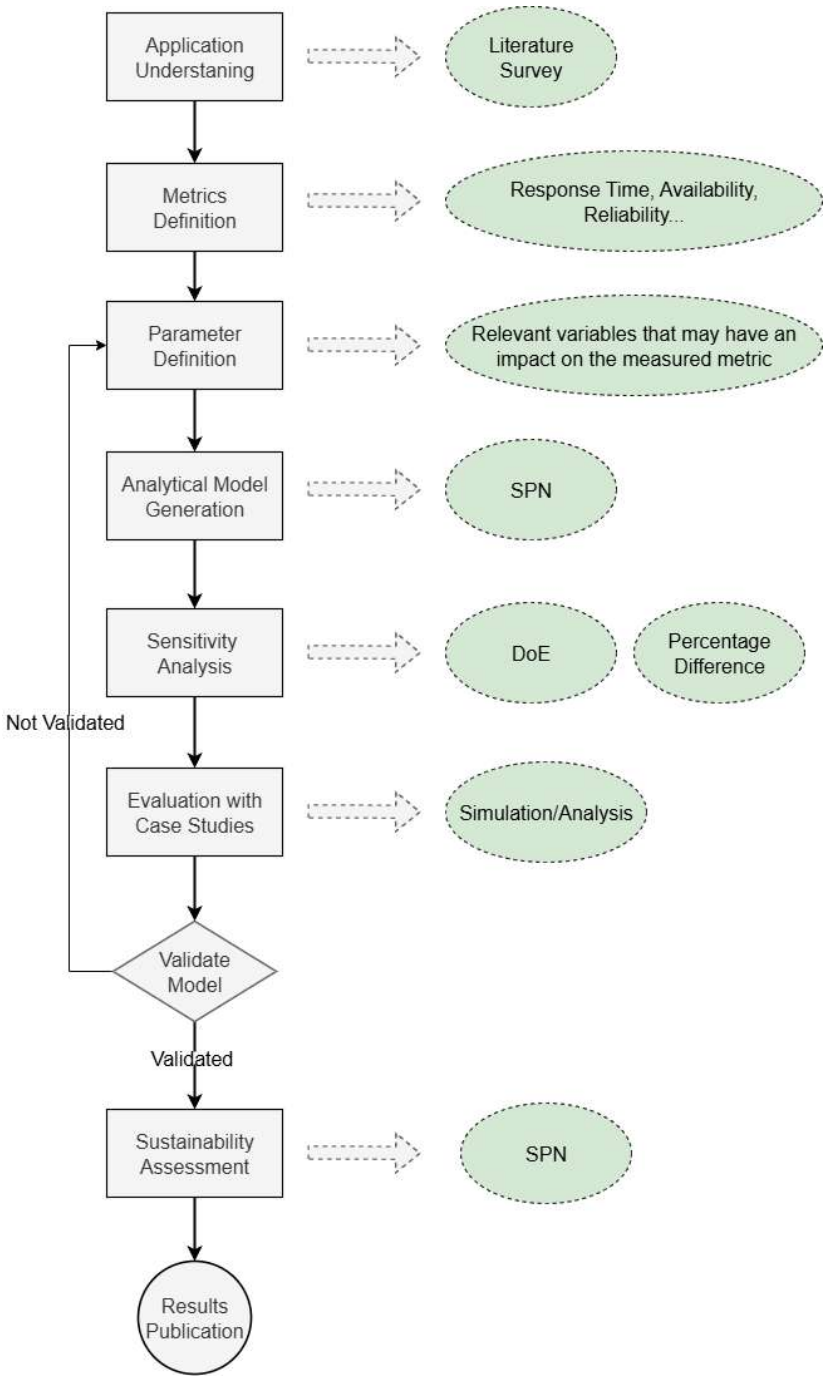


Figura 1 – Work development methodology.

- **Application Understanding:** Understanding how the application works is essential, as is identifying the number and type of components involved. In this work, this understanding was obtained through a literature review on VANETs and distributed edge architectures, which served as a basis for the subsequent modeling process.

- **Metrics Definition:** Selecting the appropriate evaluation metrics is essential to ensure that the analytical model accurately reflects the system's behavior in terms of both performance and availability under varying operational conditions. In this

work, the metrics selected were: system response time for performance analysis, in addition to complementary metrics such as utilization rate and discard rate; system availability; and the call rate per year, which will be used to evaluate the maintenance aspect.

- **Parameter Definition:** The parameters that will be inserted into the model are defined. These parameters determine the behavior and resource capacity of the components. In this work, the parameters chosen were component processing times for the performance model and failure times for each component in the availability model.

- **Analytical Model Generation:** In this step, the models are developed. Each model is built considering the metrics and parameters defined in the previous phases. The result is a performance model and models focused on the availability area. Stochastic Petri Nets are chosen because they provide the necessary and sufficient tools to describe the different behaviors and specifications of the evaluated system.

- **Sensitivity Analysis:** The application of sensitivity analysis techniques allows the system designer to understand which components are most critical. The analyses used in this work were: percentage variation and DoE. The percentage variation analysis was applied in the context of availability, while DoE was used in the context of performance.

- **Evaluation with Case Studies:** Case studies were created to verify the models' ability to generate consistent results. The case studies were also designed to serve as practical guidance, indicating how the users of the models can employ them in different analyses.

- **Validate Model:** For the validation process, the two aspects modeled in this dissertation will be used. Validation consists of taking measurements of the modeled system in reality and comparing the results with those obtained through the simulation of the models. If the results are statistically similar, the model is considered validated.

- **Sustainability Assessment:** This analysis will include checking the estimated energy consumption associated with the allocation and deallocation of virtual machines at different load levels, aiming to identify possible imbalances between performance, energy efficiency, and resource savings in multiple system configurations.

- **Results Publication:** After completing the study process and generating models, the results are disseminated through the submission of scientific articles and the writing of the dissertation.

1.5 Work Structure

The remainder of this dissertation is organized as follows: Chapter 2 presents concepts necessary to understand this dissertation; Chapter 3 presents the work related to this work; Chapter 4 describes the underlying architecture that gave rise to the proposed availability and performance models; Chapter 5 presents the part of the dissertation that performs the availability analysis of a VM migration system between RSUs, including: overview of the models, sensitivity analysis and case studies; The Chapter 6 shows the performance part of the dissertation, including: model overview, sensitivity analysis, and case studies; The Chapter 7 presents the sustainability analysis, with the energy results and the validation of the performance model. Finally, Chapter 8 presents the conclusions and final discussions on this dissertation.

2 Theorical Foudations

This chapter presents the theoretical foundations necessary for a better understanding of this work. The chapter introduces the concepts of VANETs, including some types of communication, and the main components of a VANET. Next, detailed descriptions of SPNs are presented, and their operational logic, components, and equations are discussed. Finally, the DoE and the criteria generally adopted for interpreting its results are discussed, as well as the methodological principles associated with calculating the Percentage Difference.

2.1 Vehicular Ad Hoc Networks

The accelerated process of urbanization has significantly contributed to the expansion of the vehicle fleet and, consequently, stimulated the advancement of research in VANETs, both in academia and in industry (HOTA et al., 2022; BOUKHALFA et al., 2021). This growing interest is due to the potential of VANETs to promote greater road safety, optimize traffic flow, and offer greater comfort and convenience to drivers and passengers (KASSIR et al., 2020; BOUKHALFA et al., 2021; TAHER et al., 2024). These systems face challenges arising from the high mobility of vehicles, which causes constant changes in topology and makes it difficult to maintain stable routes (MAZHAR et al., 2024). This dynamism can result in intermittent links, packet loss, and bandwidth limitations in dense scenarios, as well as vulnerabilities associated with the security and privacy of communications (ZHANG; WEI, 2021; HOZOURI et al., 2023). To mitigate these problems, VANETs seek to employ adaptive protocols, resilient routing techniques, and security mechanisms that ensure reliable communication even in highly dynamic environments.

VANETs are characterized as decentralized and self-organized wireless networks in which vehicles are able to share real-time information with each other and with infrastructure units distributed along the roads (BALZANO; STRANIERI, 2019). The application of 5G technology represented a fundamental milestone for VANETs, offering greater bandwidth and lower latency, essential for real-time communication between vehicles and infrastructure. This technology enabled the massive transfer of data, allowing the integration of critical applications (MAHMOOD; HORVáTH, 2020). In scenarios focused on road safety, for example, the detection of an accident can be quickly disseminated through the VANET, allowing vehicles in distant regions to react proactively, redirecting their routes to avoid affected areas (YU et al., 2021). The following subsections detail the main components of VANETs and the types of communication that structure their

operation.

2.1.1 Main Components of a VANET

A VANET architecture is structured from three fundamental components: The On-Board Unit (OBU), the Roadside Unit (RSU), and the Trusted Authority (TA) (AHMED; MOHAMED; SADEK, 2017). The vehicles participating in these networks are equipped with OBUs, devices responsible for enabling both direct vehicular communication (Vehicle-to-Vehicle — V2V) and communication between vehicles and infrastructure (Vehicle-to-Infrastructure — V2I) (ISMAIL et al., 2022). OBUs consist of wireless communication modules that employ dedicated technologies, such as the Dedicated Short Range Communications (DSRC) standard, based on the IEEE 802.11p and IEEE 1609 specifications (GAO et al., 2020). Additionally, they use Wireless Access in Vehicular Environments (WAVE), which enables efficient data transmission, high throughput, low latency, and reliability in communication links (MOUSA; HUSZáK; SALMAN, 2021). These embedded units process information captured by the vehicle's sensors, such as geographic position, speed, and direction, which are shared through V2V and V2I communications (SILVA et al., 2024). Such data is essential to support real-time decision-making, both by human drivers and autonomous driving systems, and to contribute to safety and efficiency in vehicular traffic.

Roadside Units (RSUs) are fixed devices installed along streets and highways, whose main function is to enable communication between vehicles and provide access to the broader network infrastructure, including central servers and cloud environments (KHAN et al., 2021). These devices can establish mutual communication through high-speed and highly reliable wired connections, in addition to maintaining dedicated links with data centers or the Internet, enabling integration with cloud services (TAHER et al., 2024; GAO et al., 2020). RSUs are capable of performing real-time processing of data received from vehicles, before forwarding them to the back-end infrastructure for further analysis (CUNHA et al., 2021; BALZANO; STRANIERI, 2019). After this step, the processed data is retransmitted by the central servers to the RSUs located in the same geographic region, which then disseminate them to the vehicles within their coverage area (HOTA et al., 2022). In addition, vehicles can use multi-hop wireless communication to propagate messages among themselves. Still, when they enter the range of an RSU, they start using direct communication with this unit (BALZANO; STRANIERI, 2019). The installation of RSUs must be carried out strategically, prioritizing critical locations in the urban network, such as intersections, parking entrances, and points of intense vehicle flow (YU et al., 2021). The efficient distribution of RSUs must seek to maximize geographic coverage and make optimal use of the available infrastructure (AHMED; MOHAMED; SADEK, 2017).

The third essential element of the VANET architecture is the Trusted Authority

(TA), which acts as a central trusted entity responsible for managing the security, integrity, and authenticity of the network (MAHMOOD; HORVáTH, 2020). The TA is considered the basis of the intelligent transportation system (ITS), since it centralizes certification, authentication, and trust control operations in highly dynamic environments (GAO et al., 2020). Its function involves the registration and management of network units, such as Roadside Units (RSUs) and On-Board Units (OBUs), as well as the issuance of digital certificates and the distribution of cryptographic parameters, including public and private key pairs to ensure the confidentiality and integrity of communications (KHAN et al., 2021). Furthermore, the TA performs continuous auditing of the network through RSUs, allowing the detection of malicious behaviors, such as the sending of spoofed packets by compromised OBUs, enabling revocation and blocking actions (AHMED; MOHAMED; SADEK, 2017). In more scalable architectures, VANETs can be organized with the support of multiple Regional Trusted Authorities (RTAs), where each RTA acts over a specific geographic region, operating subordinately to the central TA (AHMED; MOHAMED; SADEK, 2017). Typically, the main TA is located in an urban center and is managed by government authorities or regulatory bodies (KHAN et al., 2021), ensuring compliance with public policies and security regulations.

2.1.2 Types of Communication in VANETs

Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and hybrid communication are the main data transmission mechanisms in vehicular ad hoc networks (VANETs) (CUNHA et al., 2021). Each of these modalities has specific properties that make it more appropriate for different operational scenarios and applications in safety, traffic management, and infotainment. V2V communication enables the direct exchange of information between vehicles, which act as mobile nodes in a dynamic, autonomous, and decentralized network (AHMED; MOHAMED; SADEK, 2017; CUNHA et al., 2021). This type of communication is crucial for applications that require rapid response to local road conditions, such as imminent collision alerts, sudden braking, obstacle detection, and sudden changes in traffic flow. Furthermore, it enables the continuous sharing of operational data, such as geographic location, instantaneous speed, acceleration, and distance between vehicles (ISMAIL et al., 2022). V2V communication tends to present high performance in dense urban environments, but suffers significant degradation in rural areas or in scenarios with low vehicle density, where inter-vehicle connectivity becomes intermittent (KASSIR et al., 2020).

In turn, V2I communication establishes the interaction between vehicles and fixed elements of the road infrastructure, such as roadside units (RSUs), smart traffic lights, and traffic control centers (CUNHA et al., 2021). This modality is particularly effective for supporting centralized applications, such as adaptive traffic light management, electronic

toll collection, real-time map updates, and distribution of weather or traffic reports. Hybrid communication, in turn, combines the benefits of V2V and V2I, seeking to improve network resilience and coverage, especially in heterogeneous scenarios. This approach allows for greater redundancy in data transmission. It facilitates the integration between autonomous vehicles, innovative infrastructure, and urban data centers, directly contributing to the construction of smart cities and cooperative transportation systems (C-ITS). V2I acts as a complementary mechanism to V2V, being essential when the latter is unavailable, whether due to large distances between vehicles, failures in direct communication, or environments with physical obstructions and signal interference (GAO et al., 2020).

2.2 Stochastic Petri Nets

Petri Nets (PN) are a formal modeling tool, both graphical and mathematical, used to describe and analyze information processing systems with concurrent, asynchronous, distributed, and parallel behaviors (CHEN; HA, 2018). From a graphical point of view, PNs function as a visual communication support, comparable to flowcharts, block diagrams, and network diagrams, facilitating the structural understanding of complex systems. Several variations of this modeling have been developed to integrate the formal description, property verification, and performance evaluation in a single structure. Among these variations, the Generalized Stochastic Petri Nets (GSPNs) stand out, which introduce probabilistic components in order to represent the dynamics of systems more accurately (MOLLOY, 1982; GERMAN, 2000; MARSAN et al., 1994; MACIEL, 2023b; MACIEL, 2023a). For terminological simplification, such networks will be referred to here only as SPNs. In the mathematical scope, the formalism of PNs allows the formulation of state equations, algebraic systems, and other models that govern systemic behavior. Since the initial proposal by Carl Adam Petri, numerous extensions have been suggested, enabling more compact representations and the inclusion of dynamic properties not foreseen in the original versions (MURATA, 1989).

Stochastic Petri Nets (SPNs) can be formally described as bipartite directed graphs, consisting of three fundamental elements: Places, transitions, and directed arcs, which connect places to transitions and vice versa (RODRIGUES et al., 2020). Timed transitions present stochastic behavior, being triggered according to probability distribution functions (MURATA, 1989). Immediate transitions, on the other hand, are activated as soon as their enabling conditions are met and are not subject to any waiting interval. Places are represented by white circles, and arcs function as the connecting elements between places and transitions. In addition, inhibitory arcs act as control mechanisms, preventing or allowing the movement of tokens, depending on the number of tokens present in certain places. The token, in turn, is allocated to a specific place in the network, representing the current state of the system (BRITO et al., 2021). Figure 2 graphically illustrates the

constituent components of an SPN model.

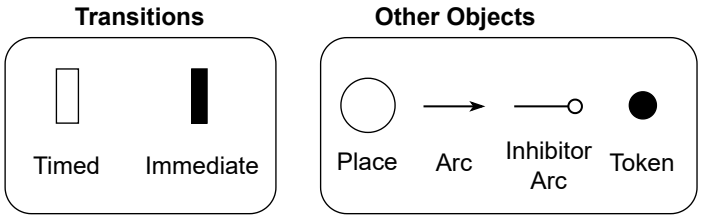


Figura 2 – SPNs components.

The behavior of an SPN is defined based on the flow of tokens between places and transitions. Tokens are created or consumed as transitions are triggered (GERMAN, 2000). Immediate transitions represent zero-duration activities and have a higher trigger priority than timed transitions. These transitions can also be associated with guard functions, allowing the specification of logical conditions for their triggering, in addition to allowing the assignment of different priority levels between them. Guard functions are Boolean expressions that regulate the triggering of transitions based on the current network marking. When the guard function associated with a transition is evaluated as accurate, the transition is enabled; otherwise, it remains disabled (MARSAN et al., 1998).

SPN models were designed with the purpose of establishing a tool capable of integrating, in a unified way, the formal description of the system, the verification of properties (correctness proof), and the performance evaluation. Figure 3 shows an SPN that models a simplified passenger boarding procedure. The token located in the check-in queue space indicates the presence of a passenger who is able to start the process. In this state, the immediate transition is enabled, and when triggered, the token is transferred to the Passenger checking in the space. The time required to execute this step varies according to the individual characteristics of the service, influenced by factors such as seat availability, the volume of baggage to be checked in, the regularity of the documentation, among others. After this interval has elapsed, the timed transition Complete check-in becomes enabled and triggers, moving the token to the Check-in completed space. As a result, and given the absence of tokens in the previous spaces, the SPN reaches a terminal state, representing the conclusion of the modeled process.

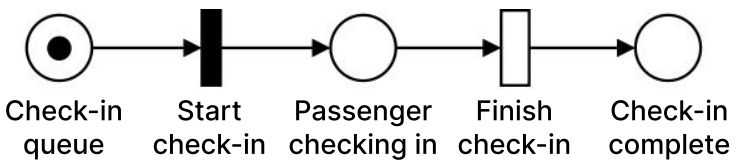


Figura 3 – Simple SPN example.

2.3 Sensitivity Analysis

Sensitivity analysis examines the impact of specific input data on results, identifying potential flaws in computational systems. Techniques are then applied to optimize these systems in different scenarios (CAMPOLONGO; TARANTOLA; SALTELLI, 1999). In general, system designers use sensitivity analysis to assess how much a metric is affected by changes in the model (SANTOS et al., 2021). There are several ways to perform this analysis, including regression analysis, perturbation analysis, Monte Carlo simulation, parametric differential analysis, correlation analysis, and percentage difference. Choosing the appropriate method can be challenging, and it is necessary to consider the available computational resources and the characteristics of the problems addressed.

Sensitivity analysis can provide the necessary security and direct the results according to the perspective previously established by the system administrators. In this work, we apply sensitivity analysis techniques based on DoE and Percentage Difference. DoE is a set of statistical techniques that improve the understanding of the product or process in question. DoE is a very effective technique for exploring new processes and gaining a deeper insight into existing processes, followed by optimizing these processes to achieve world-class performance. In the specialized literature (FEITOSA et al., 2021; COSTA et al., 2016; SANTOS et al., 2021), categories of graphs are found that are frequently used in experiments with the DoE approach. The factor effect graph, represented by bars arranged in ascending order, emphasizes the relative impact of each factor. The larger the bar, the greater its impact, allowing a clear understanding of the influences of each factor. Figure 4 presents a graph of the effect of the factors. The graph presents three factors: A, B and C. Factor C is the factor with the greatest impact.

The factor effect plot allows us to identify which interaction of factors has a more relevant impact on the optimization process or study design, indicating where attention should be directed. Understanding the interactions between factors allows us to select the best combination of measures, identifying patterns of cumulative or degrading effects between factors. The interaction between factors A and B can be estimated using the Equation (2.1 The plot {A,B(+1)} represents the effect of factor A on the highest level of factor B, while {A,B(-1)} represents the effect of factor A on the lowest level of factor B.

$$I_{A,B} = \frac{1}{2}(E_{A,B(+1)} - E_{A,B(-1)}) \tag{2.1}$$

The purpose of interaction graphs is to establish relationships between factors. An interaction occurs when the influence of one factor on the outcome is increased (or decreased) by changes in the levels of another factor. If the lines on the graph are parallel, this means that the factors are not related. If the lines are not parallel, this indicates a significant interaction between the factors in question. Figure 5a illustrates an example

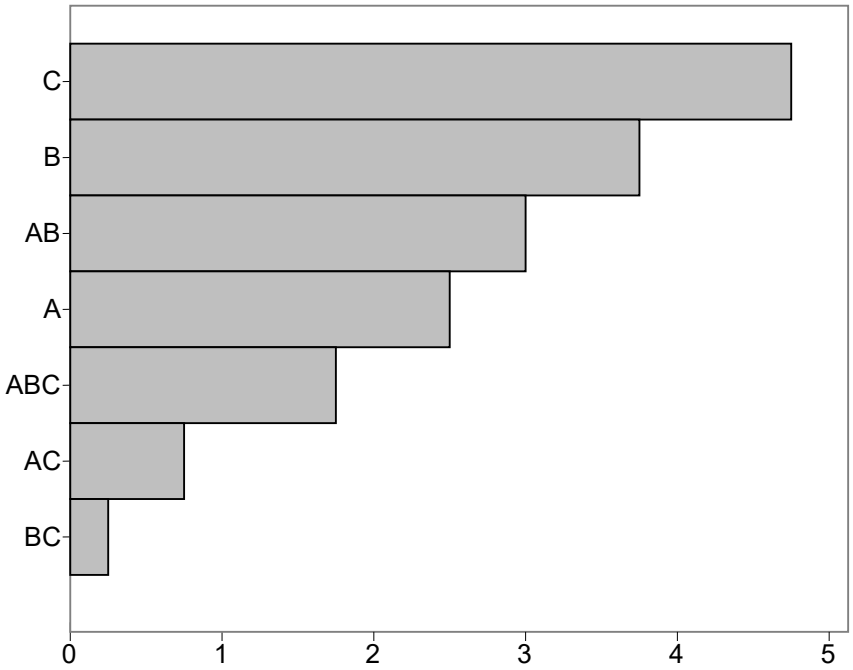
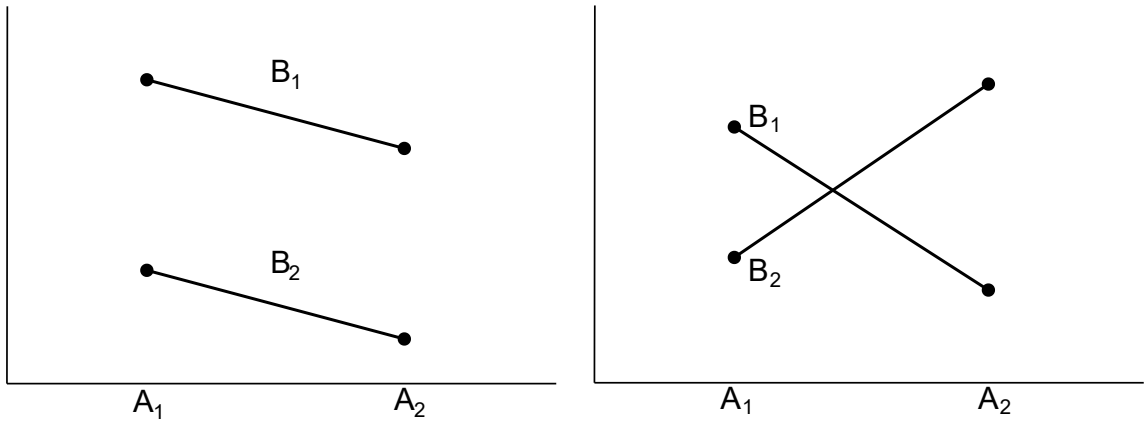


Figura 4 – Generic example of a factor effect graph.

where there is no interaction between the factors, since the lines are parallel. Figure 5b illustrates an example of interaction between factors, since the lines intersect. The change in a metric for factor A at level A1 is greater than the change at level A2. Changes in the levels of factor A for a given metric indicate a dependency between the levels of factor A and the levels of factor B.



(a) Interaction graphs with low interaction between factors.

(b) Interaction graphs with high interaction between factors.

Figura 5 – Generic interaction graphs.

3 Related Work

This chapter brings together the main studies that underpin the development of this dissertation, addressing in an integrated way the works related to performance and availability evaluation in VANETs. The first section focuses on research that employs formal modeling, queuing theories, and SPNs to represent and analyze the behavior of vehicular systems under different operating conditions. The second section expands this perspective by incorporating studies on energy efficiency and environmental impact, which explore the use of autoscaling and reinstantiation techniques focusing on energy consumption, carbon emissions, and the sustainability of distributed systems. The integration of these two sections offers a demonstration of the evolution of approaches that seek to reconcile resilience, elasticity, and energy awareness in VANET architectures. The literature has addressed resource management in VANETs using analytical methods or simulation for operational aspects. Existing works focus on latency, security, or static allocation metrics, neglecting the critical interdependence between system resilience and elasticity. Consequently, this gap is identified in formal models that articulate VM migration and dynamic automatic scaling in environments with multiple RSUs under stochastic loads.

3.1 Related Work to Performance and Availability Analysis

This section analyzes the main related work concerning monitoring and resource management in VANETs. Table 1 presents studies addressing evaluation methods, performance metrics, traffic variations, the use of multiple RSUs, types of contribution, and application contexts that were selected. The related works were identified through a systematic search of databases such as *IEEE Xplore*, *ACM Digital Library*, *Scopus*, and *SpringerLink*. The search employed combinations of keywords such as *Vehicular Ad Hoc Network*, *Stochastic Petri Net*, *performance evaluation*, *availability analysis*, *elasticity*, *autoscaling*, and *vehicular edge computing*. Only studies addressing performance, reliability, or resource management in VANETs or related vehicular edge environments were selected. This procedure ensured the inclusion of both formal modeling and simulation-based approaches relevant to the proposed analytical framework. The goal is to compare the existing proposals and highlight the gaps that motivated the development of the model proposed in this work.

Tabela 1 – Comparison of related works.

Reference	Metric	Assessment method	Traffic Variations	Multiple RSUs	Type of Contribution	Context
(TANG et al., 2019)	Transmission probability, Average delay	Mathematical Model	✓	✗	Centralized routing scheme with mobility prediction using AI and SDN	Traffic control in VANETs
(SILVA et al., 2023)	Reliability, Availability	SPN Model	✗	✓	Evaluation of VCC architectures for Urban Advanced Mobility	Availability analysis in VANETs
(CUMBAL et al., 2019)	Coverage rate, Communication rate	ILP Model	✓	✗	Efficient resource deployment in RSUs with heterogeneous communications	Dynamic coverage in vehicular networks
(WU et al., 2020)	Total Time Delay	VFC Model	✓	✗	Task assignment and resource allocation optimization	Latency reduction in VANETs
(SIDDIQI et al., 2020)	Response Time, Resource Cost, Service Quality	Queuing Model	✓	✗	Cloud integration for real-time multimedia processing	Multimedia processing in VANETs
(MARTIN-FAUS et al., 2018)	Channel Idle Time	Markov Chain Model	✓	✗	Analytical modeling of VANET behavior using Markov reward chains	Channel idle time modeling
(SHAH; ILHAN; TURELI, 2019)	Performance (MAC Layer)	Simulation	✗	✗	Performance modeling of MAC protocols	Medium access control in VANETs
(MALIK; PANDEY, 2018)	Performance (Security)	Simulation	✗	✗	Threat-oriented authentication approach	Secure communications in VANETs
(AOKI et al., 2015)	Performance (Mobile Agent Migration)	Simulation	✗	✗	Migration mechanism based on location analytics	Mobility management
(JEONG et al., 2017)	Performance (TCP Context Migration)	Simulation	✗	✓	TCP Context Migration Scheme (TOMS)	Advanced vehicular communication
(WEBER; FERRETO; ZINCIR-HEYWOOD, 2023)	Availability (Anomaly Detection)	Simulation	✗	✓	Anomaly detection and VEC techniques	Robustness analysis
(KHAN; ABBAS; HAMIDULLAH, 2019)	Performance (Container Virtualization)	Measurement	✗	✓	Container-based virtualization and live migration	Dynamic virtualization in VANETs
(GOYAL; TRIPATHI; AGARWAL, 2019)	(No performance metric)	Measurement	✗	✓	Classification of security requirements and challenges	Security analysis in VANETs

(continued on next page)

(continued from previous page)

Reference	Metric	Method	Traffic Variations	Multiple RSUs	Type of Contribution	Context
(K; CHINNASAMY, 2023)	Performance (Seamless Handover SDN)	Measurement	✗	✓	Seamless handover system using SDN and MIH	Mobility continuity in VANETs
(MCHERGUI; MOULAHI; NASRI, 2020)	Performance (BaaS Dissemination)	Measurement	✗	✓	Broadcast as a Service (BaaS) transmission using cloud computing	Data dissemination in VANETs
(ALVARENGA; SOUSA; COSTA, 2022)	Availability (Connectivity patterns)	Markov Model	✗	✗	Analytical modeling of connectivity patterns in vehicle chains	Highway vehicle connectivity
(CARDOTE; SARGENTO; STEENKISTE, 2010)	Performance (SPN VANET Infrastructure)	SPN Model	✗	✗	SPN modeling of VANET infrastructure	Mobility and connectivity analysis
(LOBO et al., 2017)	Performance (SDN Microservice Migration)	Measurement	✗	✓	Microservice allocation and migration in VFN	Vehicular fog computing
This work (2023 and 2024 Articles)	NVMU, Drop Probability, Throughput, MRT, Utilization, Availability (VM Migration)	SPN Model	✓	✓	Development of SPN models incorporating autoscaling, VM migration, and reinstantiation	Elastic resource management and availability in VANETs

Metric

The metrics adopted in the related studies reveal different evaluation priorities within vehicular environments. Some approaches focus on communication aspects, measuring successful transmission rates and average delays as performance indicators of the vehicular mesh. Others emphasize operational metrics, such as mean response time, drop probability, and resource utilization, aiming to optimize the handling of dynamic demands. Metrics related to resource cost and service quality are also used to evaluate infrastructure allocation efficiency. Despite this diversity, most works limit their analysis to isolated indicators without providing an integrated view of system behavior. Distinguishing itself from this trend, this work proposes the simultaneous analysis of multiple critical metrics — number of virtual machines in use (NVMU), drop probability (DP), throughput (TP), mean response time (MRT), and system utilization (U) — enabling the capture of both processing efficiency under varying traffic loads and operational resilience against instantiation failures and abrupt environmental changes.

Assessment method

The evaluation methods employed in the related studies reflect the diversity of strategies used to analyze vehicular systems' behavior. Analytical models based on Markov chains, queuing networks, and mathematical formulations are frequently adopted to represent the stochastic behavior of transmissions, migrations, and failure events. Large-scale simulations also emerge as an alternative to capture vehicular mobility dynamics, allowing the validation of proposals related to protocols, task allocation, or communication strategies in realistic scenarios. However, the complexity associated with vehicular networks — characterized by rapid topology changes, high density of simultaneous events, and dependence on external factors — exposes the limitations of purely analytical or simulation-based approaches. In this context, the use of SPNs stands out as a methodology that combines graphical representation, formal analysis, and the ability to model concurrency, synchronization, and probabilistic behavior. Differentiating itself from traditional approaches, this work employs SPNs to integrate, into a single model, computational elasticity through autoscaling, VM reinstantiation upon failure, and dynamic resource adaptation, enabling simulations and formal analyses that anticipate system performance across varied operational environments. Traditional approaches based on queuing theory or discrete-event simulation are limited in representing concurrent failures and adaptive behaviors under stochastic loads. Moreover, using SPNs significantly reduces experimentation costs by allowing accurate performance and availability estimations without deploying or simulating the entire physical infrastructure, thus enabling efficient evaluation of multiple scaling policies in a controlled and reproducible way.

Traffic Variations

Considering traffic variations is a decisive factor for realistically modeling the behavior of vehicular networks. Traffic dynamics in urban and highway scenarios impose intense fluctuations in processing and communication demand, which can occur over scales of minutes or hours. Some studies propose fixed scenarios or use static traffic loads, limiting the evaluation of a system's adaptive capacity to cope with demand peaks or periods of underutilization. The absence of mechanisms reflecting temporal variations in vehicle volume compromises the fidelity of the results, especially in Edge Computing architectures, where computational resources are limited and dynamic load balancing is crucial. In this work, traffic variation is explicitly modeled through scenarios with different time-based profiles, representing typical daily operational fluctuations. This approach enables the evaluation of not only the average performance but also the system's responsiveness and resilience to abrupt load changes, thereby reproducing more faithfully the operational conditions of intelligent networks in real-world environments.

Multiple RSUs

The presence of multiple RSUs in vehicular architectures is fundamental to ensuring scalability, robustness, and service continuity in distributed scenarios. In several approaches, models consider only isolated units or assume centralized communication, limiting the analysis of partial failures, load balancing, and dynamic resource reallocation. In high-mobility environments, interaction between multiple RSUs is crucial for adaptively distributing processing, minimizing delays, and ensuring tolerance to local failures. Models that disregard this multiplicity tend to underestimate operational bottlenecks and overload specific units during demand peaks. This work advances beyond these limitations by incorporating distributed resource management among RSUs, considering the ability to instantiate, migrate, and reinstate virtual machines across different units based on traffic conditions and computational availability. This approach provides a more realistic view of the operational challenges faced by intelligent traffic infrastructures, enabling more accurate assessments of large-scale performance and availability.

Type of Contribution

The contributions observed in the related studies reveal a diversity of approaches, ranging from the proposal of new communication protocols, optimization strategies for resource allocation, anomaly detection mechanisms, to security improvements for vehicular networks. In general, these contributions focus on addressing isolated problems, such as reducing latency, expanding network coverage, or mitigating security failures, without necessarily integrating multiple aspects of performance and availability into a unified solution. Furthermore, many works propose solutions based solely on simulations, without the support of formal models that allow mathematically rigorous analyses of expected behaviors. Distinguishing itself from this scenario, the present work proposes a more comprehensive contribution by integrating capacity autoscaling mechanisms, virtual machine instantiation after failures, and dynamic adaptation to traffic variability into a single stochastic model. This approach enables not only the evaluation of performance under different operational conditions but also the anticipation of critical system behaviors, providing more robust support for planning intelligent vehicular infrastructures.

Context of Availability and Performance Analysis

Several related studies focus on traffic control and network coverage in VANETs. (TANG et al., 2019) presents a centralized routing model using AI-based mobility prediction to improve transmission probability and reduce delay. (CUMBAL et al., 2019) proposes an ILP model to optimize heterogeneous vehicle coverage by integrating technologies such as DSRC and LTE. (WU et al., 2020) addresses dynamic task assignment and

resource allocation in vehicular fog computing networks to minimize total response time. Although contributing to communication performance, these works do not incorporate computational elasticity or resilience mechanisms as proposed in this work. In the context of edge computing, (SIDDIQI et al., 2020) models real-time multimedia data processing by integrating vehicles with the cloud, focusing on response time, resource cost, and service quality. (BRITO et al., 2020) applies Stochastic Petri Nets to evaluate mean response time, drop probability, and utilization in edge RSUs, emphasizing resource underutilization scenarios. (MARTIN-FAUS et al., 2018), models channel idle time with Markov reward chains to optimize medium access protocols. These works address efficiency but do not integrate autoscaling or service reinstantiation strategies under dynamic and failure conditions.

Some works focus on medium access and security improvements. (SHAH; ILHAN; TURELI, 2019) analyzes MAC layer performance in high-density scenarios. (MALIK; PANDEY, 2018) proposes a threat-oriented authentication scheme, while (GOYAL; TRIPATHI; AGARWAL, 2019) classifies security requirements and challenges in VANETs. These proposals, while relevant, are limited to specific communication layers and do not address system-wide resource elasticity or failure recovery mechanisms. Regarding information dissemination and agent migration, (HASHIMOTO et al., 2015) proposes a mobile agent scheme for contextual information sharing, and (AOKI et al., 2015) introduces a location-based migration mechanism to reduce latency. (JEONG et al., 2017) develops the TCP Context Migration Scheme (TOMS) to ensure session continuity during handovers. Although important for communication mobility, these works do not model computational elasticity or dynamic service reallocation.

Focusing on network robustness, (WEBER; FERRETO; ZINCIR-HEYWOOD, 2023) applies conventional and VEC-based techniques for anomaly detection and message loss mitigation. Meanwhile, (KHAN; ABBAS; HAMIDULLAH, 2019) proposes a container-based virtualization architecture supporting dynamic migration in VANETs. These approaches enhance service reliability but do not formally address service reinstantiation or resource autoscaling in multi-RSU systems. Other contributions aim to ensure connectivity, continuity, and manage microservices. (K; CHINNASAMY, 2023) proposes a seamless handover system based on SDN and MIH, while (MCHERGUI; MOULAHI; NASRI, 2020) introduces Broadcast as a Service (BaaS) for efficient data dissemination through cloud platforms. (ALVARENGA; SOUSA; COSTA, 2022) explores microservice allocation and migration in vehicular fog networks using SDN. However, these works do not explicitly incorporate resource elasticity or resilience modeling for distributed infrastructures. Formal modeling efforts include (CARDOTE; SARGENTO; STEENKISTE, 2010), which uses Markov chains to analyze connectivity patterns in vehicle chains, and (LOBO et al., 2017), which models VANET infrastructures using Stochastic Petri Nets to capture mobility and connectivity degradation impacts. While advancing analytical

modeling, these studies lack integration with dynamic resource management and fault recovery strategies, areas addressed by this work.

Distinguishing itself from previous approaches, this work proposes a formally modeled architecture that integrates, within a single analytical framework, the joint evaluation of performance and availability in RSU-based vehicular systems. By employing Stochastic Petri Nets, the developed model incorporates computational elasticity mechanisms through autoscaling, virtual machine reinstantiation in response to service failures, and dynamic load redistribution across multiple units. In addition to capturing temporal traffic variations and the impact of mobility on computational resources, the model enables formal simulations that anticipate critical behaviors under different operational scenarios. This approach offers an unprecedented technical planning capability for intelligent vehicular infrastructures, allowing proactive identification of bottlenecks, robustness evaluation against failures, and early optimization of resource allocation—dimensions that remain underexplored in the related studies.

3.2 Related Works to Sustainability Analysis

This section presents an analysis of related works, focusing on studies that address load balancing and/or energy consumption in VANETs. The works are classified into two groups based on the evaluation method employed: (i) Simulation-Based Analysis in VANETs and (ii) Model-Based and Simulated Approaches to VANETs. Table 2 presents the main characteristics of these studies. The works were identified through a systematic search of databases such as *IEEE Xplore*, *ACM Digital Library*, *Scopus*, and *SpringerLink*. The search employed combinations of keywords such as *Vehicular Ad Hoc Network*, *energy consumption*, *energy efficiency*, *load balancing*, *autoscaling*, and *resource management*, aiming to identify studies addressing energy-aware optimization in *fog* and *edge* environments. The retrieved studies were filtered and classified according to the adopted evaluation method, *simulation-based analysis* or *model-based and simulated approaches*, allowing a structured.

Tabela 2 – Related sustainability works

Work	Metrics	Evaluation Method	Energy	Autoscaling	Type of Contribution	Focus
(RAZA et al., 2021)	Packet delivery rate, packet latency, avg. energy consumption	Simulation	✓	✓	Empirical evaluation via simulation	Communication
(RAJENDRAN et al., 2023)	Throughput, packet delivery rate, comm. delay, comm. overhead	Simulation	✗	✓	Protocol performance assessment	Communication

(continued on next page)

(continued from previous page)

Work	Metrics	Evaluation Method	Energy	Autoscaling	Type of Contribution	Focus
(MOHAMMED et al., 2023)	Packet delivery rate, end-to-end delay, routing overhead, throughput, energy consumption	Simulation	✓	✓	Routing/energy trade-off study	Communication
(MEHTA; MAHATO, 2023)	Energy consumption, queue wait time, load balancing efficiency	Simulation	✓	✓	Load-balancing with energy awareness	Energy efficiency
(SETHI; PAL; VYAS, 2020)	Energy consumption, % of requests served	Simulation	✓	✓	Resource management policy analysis	Energy efficiency
(WU et al., 2022)	Task response time, vehicle energy, load balancing	Simulation, modeling	✓	✓	Autoscaling optimization with models	Autoscaling optimization
(HAIDER; KHAN; SAEED, 2024)	Residual energy, end-to-end delay, delivery rate	Simulation	✓	✓	Heuristics for energy–delay balance	Autoscaling optimization
(WU; OBAIDAT; CHAN, 2015)	Workload rate, latency, load balance index	Simulation, modeling	✗	✓	Workload-aware scaling strategies	Communication
(QUN; AREF-ZADEH, 2021)	Total energy, network lifetime, imbalance degree	Simulation, modeling	✓	✓	Topology/energy co-optimization	Energy efficiency
(AGARWAL; DAS; DAS, 2016)	Energy consumption, network load, avg. packet delay	Simulation, modeling	✓	✓	Network/energy joint evaluation	Energy efficiency
(ALI et al., 2014)	Transmission time, packet loss, response time, throughput	Simulation	✗	✓	Traffic/throughput analysis	Communication
(ALI; CHAN; LI, 2014)	Deadline miss, stretch, delivery success	Simulation, modeling	✗	✓	SLA-aware scaling policies	Autoscaling optimization
(VIJAYAKUMAR; JOSEPH, 2019)	Deadline miss, queue time, throughput, delivery rate	Simulation	✗	✓	Queue-aware dissemination evaluation	Communication
This Work	Energy consumption, carbon footprint, energy efficiency, response time	Modeling	✓	✓	SPN-based sustainability modeling	Sustainability

Simulation-Based Analysis in VANETs

The studies (RAZA et al., 2021), (RAJENDRAN et al., 2023), (MOHAMMED et al., 2023), and (MEHTA; MAHATO, 2023) used Network Simulator 2 (NS2) as the primary simulation tool. The study (RAZA et al., 2021) proposed an architecture that integrates UAVs to balance network traffic in congested areas, using NS2 and BonnMotion. The study (RAJENDRAN et al., 2023) developed a routing model to improve communication quality in distributed VANETs, evaluating different traffic scenarios using NS2. The study (MOHAMMED et al., 2023) focused on resource allocation and energy balancing in

UAV-assisted networks, aiming to reduce latency and communication overhead through simulations in NS2. Finally, the study (MEHTA; MAHATO, 2023) proposed a hybrid load balancing scheme, minimizing energy consumption in RSUs through an efficient request scheduling algorithm, evaluating its performance in NS2, SUMO, and OpenStreetMap.

On the other hand, the studies (SETHI; PAL; VYAS, 2020), (HAIDER; KHAN; SAEED, 2024), (ALI et al., 2014), and (VIJAYAKUMAR; JOSEPH, 2019) used different simulation tools. The study (SETHI; PAL; VYAS, 2020) presented an energy-efficient scheduling algorithm for RSUs powered by renewable energy sources, distinguishing between traditional and intelligent applications, with simulations conducted in NS3, SUMO, and OpenStreetMap. The study (HAIDER; KHAN; SAEED, 2024) proposed an adaptive load balancing mechanism to mitigate congestion in VANETs, organizing RSUs into clusters and managing route selection and request scheduling, evaluated using VanetMobiSim with NS2 extension and CANU Mobility Simulation Environment. The study (ALI et al., 2014) developed a cooperative approach to load balancing in RSUs, improving communication efficiency between vehicles and infrastructure, evaluated in CSIM19, demonstrating a reduction in the rate of expired requests. Finally, the study (VIJAYAKUMAR; JOSEPH, 2019) proposed an adaptive load balancing scheme (ALB) to optimize data dissemination in VANETs within the V2I model, also evaluated in CSIM19, resulting in improvements in delivery success rate and communication reliability. Although these simulation-based approaches offer valuable insights into VANET behavior, they often lack formal models capable of capturing system dynamics under autoscaling policies and energy constraints.

Model-Based and Simulated Approaches to VANETs

The studies (WU et al., 2022) and (QUN; AREFZADEH, 2021) utilized bio-inspired algorithm-based modeling to optimize load distribution in VANETs. The study (WU et al., 2022) proposed a vehicular computation offloading scheme with guaranteed load balancing, employing Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC) to model the dynamic task allocation within the network. Similarly, the study (QUN; AREFZADEH, 2021) used a hybrid ACO-ABC algorithm to optimize energy efficiency and load distribution in fog-based VANETs. Both approaches were evaluated through simulations in NS2, demonstrating improvements in load distribution and energy consumption.

Alternatively, the studies (WU; OBAIDAT; CHAN, 2015), (AGARWAL; DAS; DAS, 2016), and (ALI; CHAN; LI, 2014) applied statistical modeling to predict and balance the load among network elements. The study (WU; OBAIDAT; CHAN, 2015) introduced QualityScan, a scheme based on Little's Law to estimate the load on access points (APs) and optimize user allocation, validated through simulations in NS2. The study (AGARWAL; DAS; DAS, 2016) proposed a mathematical model based on packet

arrival and transmission rates, exploring different node distances to minimize energy consumption, with validation conducted in MATLAB. Meanwhile, the study (ALI; CHAN; LI, 2014) employed statistical methods to predict the future load of RSUs, ensuring efficient request balancing, with validation performed through simulations in CSIM19. Despite the advancements in modeling and hybrid methods, many studies rely on abstract assumptions or do not validate their models against real deployments, which limits their practical applicability.

Context of Sustainability Analysis

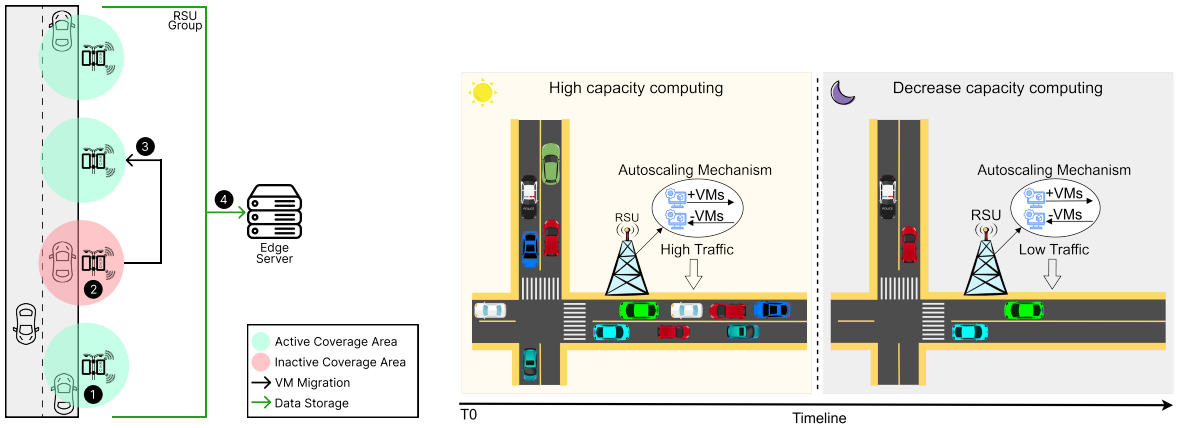
Unlike existing approaches in the literature, our methodology exploits the flexibility of SPNs, offering a probabilistic and structured model for energy-efficient load balancing in VANETs. Our approach dynamically adjusts the computational resources of RSUs by integrating autoscaling systems, reducing energy consumption, and carbon emissions. This adaptability ensures that resources are allocated efficiently according to network demand, minimizing underutilization and overload scenarios. In addition, we employ the DoE methodology to perform sensitivity analyses, which allow a quantitative assessment of the main parameters that influence the system's sustainability.

This methodological improvement enables more accurate decision-making, promoting a scalable and environmentally sustainable vehicular infrastructure. MRT was used as a performance indicator to evaluate the system's responsiveness under different load levels. Finally, it is worth highlighting that the proposed model was validated through practical experimentation with the Pasid-Validator tool, demonstrating statistical adherence between the simulated behavior and the empirical data, which reinforces the reliability and applicability of the model in real scenarios.

4 Architecture

This chapter presents the proposed architecture, whose analytical modeling is detailed in the following chapters using SPNs. The architecture is based on two mechanisms applied independently: The first ensures operational continuity through the migration of virtual machines between RSUs, while the second dynamically adapts the computing capacity through horizontal autoscaling and VM reinstantiation. Figure 6 presents the architectures adopted for the development of the models presented in the following chapters. This is the conceptual architecture, whose purpose is to abstract the organization and operational flow between the system components.

This approach aims to design, in a formal and technology-independent manner, the resilience and adaptation strategies that will be analyzed using Stochastic Petri Nets. The architecture was divided into two parts for better understanding. Although the migration and autoscaling mechanisms operate independently and are activated by different events, physical or logical failure in the first case, load variation in the second, the architecture proposed in this chapter presents them in an integrated manner, with a view to providing a cohesive view of the adaptive and resilient strategies applied to the VANETs domain.



(a) Architecture adopted for the availability model.

(b) Architecture adopted for the performance model.

Figura 6 – Architecture adopted for SPN models.

Figure 6a contains the first system incorporated into the architecture responsible for ensuring service continuity in the event of physical or logical failures in the RSUs. This component is based on detecting unavailability in an RSU and the consequent migration of its virtual machines to another active unit in the network. The architecture is composed of multiple RSUs interconnected and synchronized with an edge server, which

is responsible for storing and consolidating the collected data. The operational dynamics of this architecture are as follows:

- Active RSU Coverage and Vehicle Interaction: Vehicles in transit enter the coverage area of active RSUs (represented by green circles), where these RSUs facilitate communication and collect data from the vehicles.
- RSU Failure Response: In the event of a malfunction of an RSU, leading to an interruption in data collection, a contingency protocol is activated.
- VM Migration for Uninterrupted System Availability: To ensure continuous system functionality, a virtual machine (VM) data allocation is performed that will be transferred to the subsequent RSU within the network upon the failure of an RSU.
- Data Management: After collection, all data is transmitted to an Edge Server for storage and further processing.

Failure detection is conducted locally on each RSU through internal monitoring, and once the failure is confirmed, the migration process of the associated VMs begins. This procedure ensures that running services are not interrupted, promoting load redistribution among the available RSUs, with minimal impact to vehicles in transit. The corresponding availability model that will be presented in the chapter 5, considers failure and recovery transitions, as well as the operational state of logical, physical, network, and edge components, ensuring traceability of the effects of migration on the overall operation of the system.

Figure 6b illustrates its structure, focusing on the dynamic adaptation of highway monitoring. This architecture served as the basis for the development of the performance model, contained in chapter 6. The system automatically adjusts computational resources according to traffic variations throughout the day or times of heavy traffic, ensuring operational efficiency. The main difference is the autoscaling mechanism of the RSUs, which modifies the processing capacity according to demand. Two main factors regulate the allocation of VMs: The volume of requests generated by vehicles on the highway and the processing load of the RSUs. This control prevents resource waste and overload, optimizing system performance. The operation of the RSUs dynamically adapts to the detected traffic flow.

During periods of low traffic, the RSU operates at reduced capacity, maintaining only essential VMs. As the number of vehicles increases, the system continuously monitors the processing load and identifies possible computational bottlenecks that may compromise performance. When an overload threshold is reached, the autoscaling mechanism is automatically triggered, instantiating new VMs to expand processing capacity. This

adjustment occurs continuously and without the need for manual intervention. Along with the autoscaling mechanism, there is also a VM reinstantiation mechanism.

This mechanism seeks to ensure the correct instantiation of VMs, which will be discussed in more detail later. For example, at night, when traffic is reduced, the system deactivates excess VMs to save computing resources, maintaining only the capacity necessary to meet the minimum demand. The flow of vehicles increases significantly during the day, especially during peak hours, requiring greater processing capacity. At these times, autoscaling dynamically allocates more VMs to handle the increasing computational load, ensuring that all requests are served without performance degradation. The approaches described support variations in the number of edge units, allowing them to adapt to different traffic densities and urban settings.

This flexibility is guaranteed by the independence of the RSUs, which operate autonomously in their domains and are interconnected only when necessary, as in the case of VM migration. The ability to scale locally and migrate globally allows the architecture to be adjusted to scenarios with high traffic variability and contexts subject to localized failures. In both cases, the proposed mechanisms preserve service continuity and avoid overload or abrupt interruptions in the processing of vehicular data. This characteristic makes the architecture applicable to different deployment strategies, including urban stretches with heavy traffic and road segments with distributed infrastructure subject to intermittent degradation.

While approaches based on physical redundancy or service replication offer traditional ways to maintain operational continuity in failure or overload scenarios, they often imply high acquisition and operation costs. Unlike approaches based on physical redundancy, the architecture described here adopts strategies aimed at maintaining operation through logical reconfiguration and resource redistribution. These mechanisms are formalized in the analytical models developed in the following chapters, which independently represent the behaviors associated with migration and scheduling in distributed vehicular systems. The results were obtained using the Mercury tool (version 5.0.1) [1] (MACIEL, 2023a).

[1] https://www.modcs.org/

5 Availability Analysis

This chapter presents the proposed availability architecture, the developed availability model using the Mercury tool, the sensitivity analysis conducted through experimental design, and the case studies developed to evaluate the system performance. The proposal aims to analyze the resilience of VANETs in the face of failures and dynamic operational scenarios, especially when integrated with virtual machine migration mechanisms in edge environments. The architecture considered comprises multiple RSUs, edge servers, and failure detection mechanisms with recovery strategies based on VM migration. These elements collaborate to maintain service continuity and mitigate the impacts of occasional outages, ensuring the integrity of the vehicular data flow.

The model takes into account both the physical and logical availability of the RSUs, in addition to the average failure and recovery times of each component. To evaluate the system's availability, models based on SPNs are used, with simulations that incorporate multiple failure and recovery scenarios. Sensitivity analysis, conducted using factorial planning, allows us to identify the parameters with the most significant impact on global availability, supporting architectural and maintenance adjustments. Finally, the case studies demonstrate, in a practical way, the effectiveness of the proposed models, highlighting the gains provided by VM migration in the resilience of urban vehicular networks.

5.1 Models Overview

This section describes the models applied in this chapter. It details the representative availability model in scenarios with and without the implementation of migration strategies, and the reliability model.

5.1.1 Availability Model With Migration

Table 3 systematically details the elements of the availability model, employing tokens to represent the count of operational Virtual Machines (VMs) within each Road Service Unit (RSU). In this model, the transitions labeled RSU_MTTF (Mean Time To Failure) and RSU_MTTR (Mean Time To Repair) are integral for facilitating the exchange of information among different RSU groups via VMs. For the effective operation of an RSU Group, the quantity of tokens in the RSU_UP state must align with the initially set value.

The availability of an RSU is contingent upon the presence of tokens in the RSU_LOG_UP state and their absence in the RSU_LOG_DW state. Concurrently, the

operational state of the NETWORK is indicated by the distribution of tokens: tokens in the NET_UP state signify an active network, while those in NET_DW denote an inactive state. This model underscores the reliance on token distribution to depict the dynamic status of each RSU and the overall network, thus providing a comprehensive view of the system's availability and the efficacy of the migration strategy.

Tabela 3 – Description of the main components of the model.

Type	**Components**	**Description**
Places	MIGRATE_UP	Migration of VM's between RSU's is available
	MIGRATE_DW	Migration of VM's between RSU's is unavailable
	EDGE_UP	Data processing at the edge is available
	EDGE_DW	Data processing at the edge is unavailable
	NET_UP	The network connecting the RSU's is available
	NET_DW	The network connecting the RSU's is unavailable
	RSU_UP1, RSU_UP2, RSU_UP3	The physical RSU's are available
	RSU_DW1, RSU_DW2, RSU_DW3	The physical RSU's are unavailable
	RSU_LOG_UP1, RSU_LOG_UP2, RSU_LOG_UP3	The logical RSU's are available
	RSU_LOG_DW1, RSU_LOG_DW2, RSU_LOG_DW3	The logical RSU's are unavailable
Transitions	E_MTTR	Represents the MTTR of the system's Edge computing
	E_MTTF	Represents the MTTF of the system's Edge computing
	NET_MTTR	Represents the MTTR of the system's network
	NET_MTTF	Represents the MTTF of the system's network
	RSU_MTTR1, RSU_MTTR2, RSU_MTTR3	Represents the MTTR of the system's RSU's
	RSU_MTTF1, RSU_MTTF2, RSU_MTTF3	Represents the MTTF of the system's RSU's
	LOG_MTTR1, LOG_MTTR2, LOG_MTTR3	Represents the MTTR of the logical part of the RSU's
	LOG_MTTF1, LOG_MTTF2, LOG_MTTF3	Represents the MTTF of the logical part of the RSU's
	T1, T2, T3, T4	Transitions between RSU's in the system

The transitions labeled NET_MTTF (Mean Time To Failure) and NET_MTTR (Mean Time To Repair) are pivotal in the management of the network's operational dynamics. Concurrently, the functioning of the EDGE component relies on the E_MTTF and E_MTTR transitions, which govern the activation (EDGE_UP) and deactivation (EDGE_DW) states. The migration process, a key element for ensuring uninterrupted system operation, is regulated through MIGRATE_UP (activation) and MIGRATE_DW (deactivation) markers. These markers facilitate the transfer of Virtual Machines (VMs) between RSUs, an action critical for maintaining system availability.

Figure 7 presents the proposed SPN model, which is constructed based on the previously outlined scenario. This model encompasses various components such as RSU GROUP, RSU LOGICAL GROUP, NETWORK, EDGE, and MIGRATION. Each of these components is associated with metrics like MTTF and MTTR, which are instrumental in evaluating the availability and reliability of systems and their components.

The operational status of the Virtual Machines (VMs) in each RSU is indicated by the presence of tokens in the RSU_UP and RSU_DW states. Here, RSU_UP denotes active RSUs, while RSU_DW represents inactive ones. The transitions between the active and inactive states of each Road Service Unit (RSU) are controlled by the RSU_MTTF and RSU_MTTR transitions. Within the RSU Group, each unit contributes to the collective functionality by sharing information via Virtual Machines (VMs). For the RSU Group to function effectively, it is essential that the number of tokens in RSU_UP aligns with its

pre-established initial value.

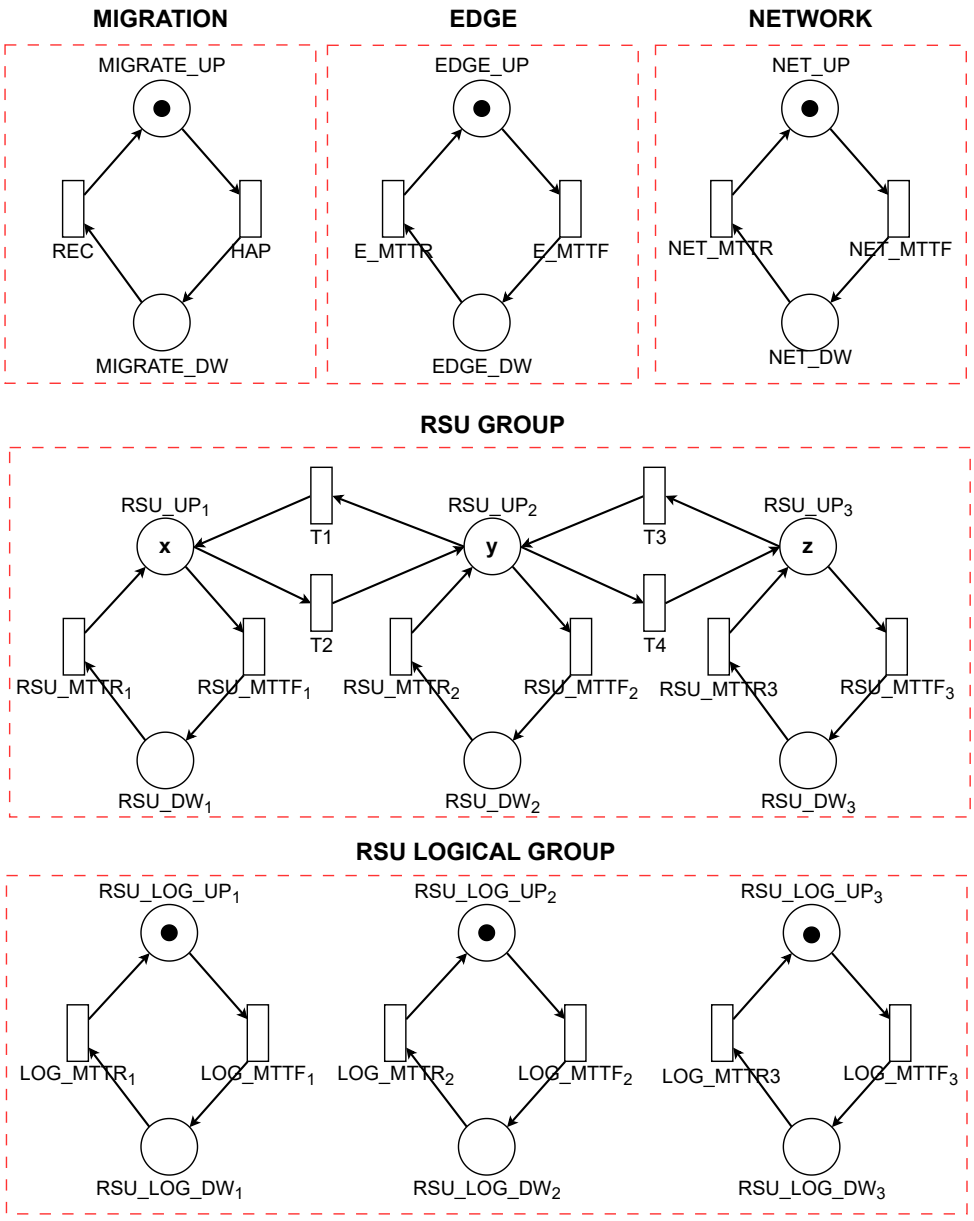


Figura 7 – Availability model with migration

Table 4 illustrates the implementation of guard conditions in the transitions T1, T2, T3, and T4, which are situated between the Road Service Units (RSUs) designated as UP1, UP2, and UP3. These guard conditions are essential for regulating the migration of Virtual Machines (VMs) in scenarios where a logical component of the Road Service Unit (RSU) becomes unavailable. When this logical component is subsequently reactivated and resumes regular operation, the previously migrated VMs are reintegrated into their original RSU. This mechanism of migration and reintegration is pivotal in ensuring the seamless and continuous functioning of the system, as it provides a dynamic response to temporary outages or disruptions within individual RSUs, thereby maintaining overall system integrity and operational continuity.

Tabela 4 – Description of model storage conditions

Places	Transition	Condition
RSU_UP1, RSU_UP2	T2	IF(#RSU_LOG_DW2=0): (N-(#RSU_UP1+#RSU_DW1)) ELSE(#RSU_UP1+(#RSU_UP1+#RSU_UP2+#RSU_UP3))
RSU_UP2, RSU_UP3	T4	IF(#RSU_LOG_DW3=0): (N-(#RSU_UP2+#RSU_DW2)) ELSE(#RSU_UP2+(#RSU_UP1+#RSU_UP2+#RSU_UP3))

The transition T2 is activated by the guard condition #RSU_LOG_Dw1=1, which denotes a critical event indicative of the instability or inactivation of RSU 1's logical component. This event initiates the process for virtual machine (VM) migration from the compromised RSU. The procedure begins by verifying the operational status of the subsequent RSU, indicated by #RSU_LOG_Dw2=0. Should this RSU be operational, the migration of VMs is executed accordingly. Subsequently, in the event of a recovery and reactivation of the initially failed RSU, the previously migrated VMs are reintegrated into it. In a parallel scenario, transition T4 is invoked when #RSU_LOG_Dw2=1, a condition signaling the deactivation of the logical component of RSU 2. This state necessitates the migration of VMs from RSU 2 to another operational unit, thereby ensuring the continuity of system functionality.

The availability of the model with migration is quantified using Equation 5.1. This equation computes the probability that the RSU Group, RSU Logical Group, Network, and Edge components are all operational concurrently. In this context, 'P' denotes the likelihood, while 'TOKENS' refers to the number of tokens present in a specific state or place within the model. This approach to calculating availability is crucial for assessing the effectiveness of the migration strategy in maintaining the continuous operation of the system, even in the face of individual component failures or disruptions. The inclusion of these probabilistic measures provides a comprehensive understanding of the system's resilience and its ability to sustain uninterrupted service through dynamic VM migration processes.

$$
\begin{aligned}
A = P((\#EDGE_U > 0) \text{ AND } (\#NET\_UP > 0) \text{ AND } \\
((\#RSU\_UP1 + \#RSU\_UP2) + \#RSU\_UP3) = (n \cdot \text{TOKENS}) \text{ AND } \\
(\#RSU\_LOG\_UP1 = 1 \text{ OR } \#RSU\_LOG\_UP2 = 1 \text{ OR } \#RSU\_LOG\_UP3 = 1))
\end{aligned}
\tag{5.1}
$$

5.1.2 SPN Availability Model: Non-Migration Framework

The depicted SPN availability model in Figure 8 delineates the system's functionality in the absence of migration capabilities. This model parallels its counterpart, incorporating migration, encompassing components such as MIGRATION, EDGE, NETWORK, RSU GROUP, and RSU LOGICAL GROUP. Integral to each component is an MTTF and a

singular MTTR, with the exception of MIGRATION, which is characterized by the Happened attribute (HAP). This attribute simulates potential disasters or system instabilities, leading to the activation of MIGRATE_DW, thereby deactivating migration processes.

A notable deviation in this non-migratory model is the omission of transition mechanisms within the RSU GROUP, which precludes inter-RSU migration. The activation of the recovery process (REC) is contingent upon the RSU GROUP's configuration, which facilitates the identification of operational RSUs via the tokens in RSU_UP. The symbol 'N' signifies the possibility of multiple tokens within each RSU, with the quantity of tokens representing the number of operational RSUs.

Conversely, the RSU_DW's token count reflects the number of non-operational RSUs. The transitions RSU_MTTF and RSU_MTTR govern the oscillation between active and inactive states of individual RSUs. Network functionality is ensured when a token resides in NET_UP (active Network), and it becomes non-functional with a token in NET_DW (inactive Network). The transitions NET_MTTF and NET_MTTR regulate these state changes.

Similarly, the cloud's operational status is indicated by the presence of a token in EDGE_UP, and its non-operational status is signified by a token in EDGE_DW. The transitions EDGE_MTTF and EDGE_MTTR are instrumental in toggling between the cloud's active and inactive states.

5.1.3 System Reliability Model

The model for analyzing the reliability of Vehicular Communication Systems (VANETs) is depicted in Figure 9. Within this context, reliability is defined as the conditional probability that a system will continue functioning over a time interval [0, t], provided that it was operational at the inception of this interval (t = 0) [24]. The presented model (Figure 9) bears resemblance to the model in Figure 7, with a notable distinction: it excludes the MTTR transitions that would facilitate the recovery of components in the event of a failure. This exclusion is a critical aspect as it directly impacts the system's ability to self-recover post-failure, thereby influencing the overall reliability assessment of the VANET system under study.

In the RSU GROUP within the VANET system, the RSUs can host varying numbers of VMs, symbolized by the variables x, y, and z. The reliability of the model under consideration can be quantitatively assessed using Equation 6.1. This metric is defined as the complement of the probability of failure of any component in the system. Specifically, it is represented as one minus the probability that the RSU Group, RSU Logical Group, Network, and Edge components perform all operations simultaneously.

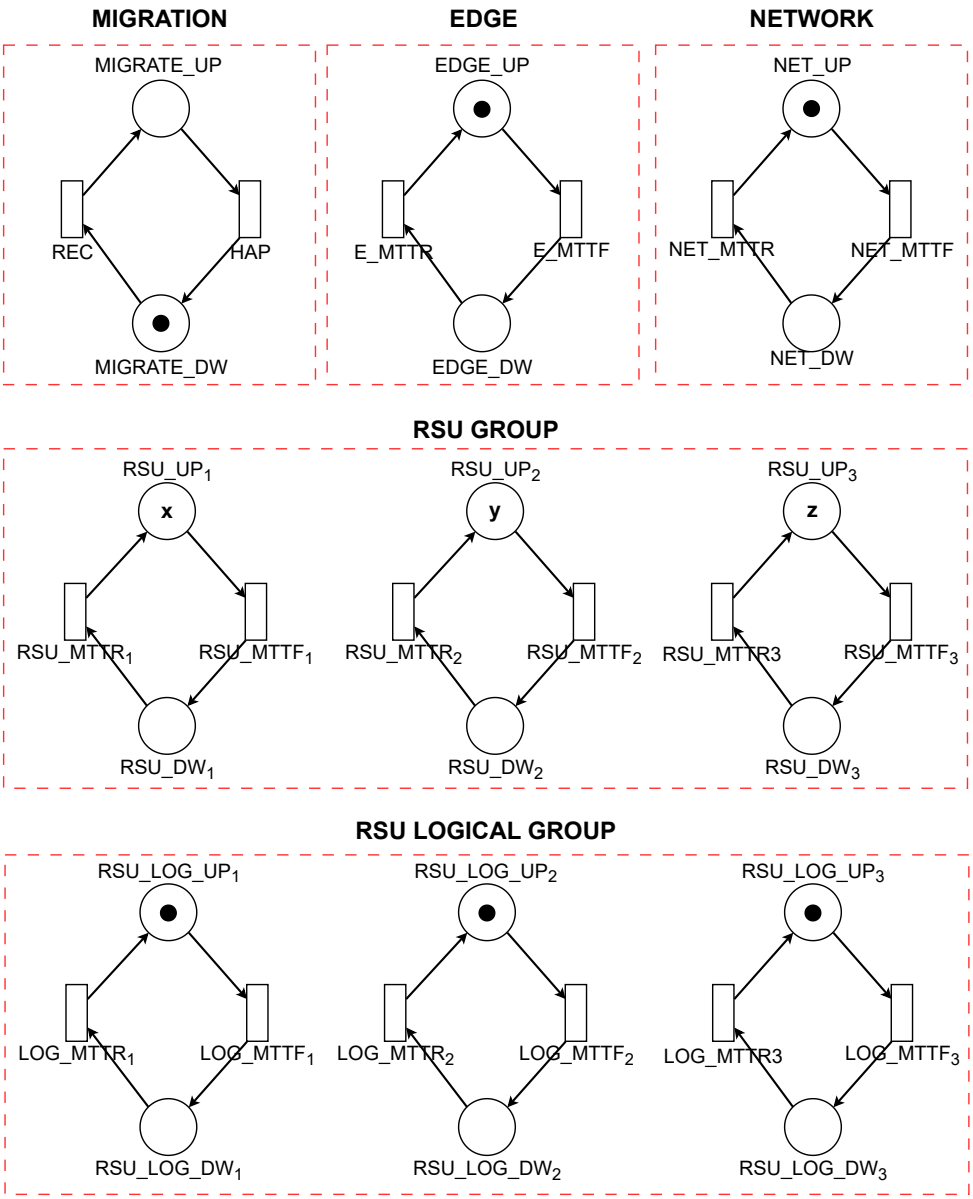


Figura 8 – Availability model without migration

$$
\begin{aligned}
R = 1 - P((\#EDGE_U > 0) \text{ AND } (\#NET\_UP > 0) \text{ AND } \\
((\#RSU\_UP1 + \#RSU\_UP2) + \#RSU\_UP3) = (n \cdot \text{TOKENS}) \text{ AND } \\
(\#RSU\_LOG\_UP1 = 1 \text{ OR } \#RSU\_LOG\_UP2 = 1 \text{ OR } \#RSU\_LOG\_UP3 = 1))
\end{aligned}
\tag{5.2}
$$

In this model, the variable "P"is utilized to determine the probability that the system will become unavailable or fail. By applying this equation, it is possible to generate a curve that effectively illustrates how the system's reliability diminishes over time. This curve is a crucial analytical tool as it provides a visual representation of the system's reliability, enabling the identification of trends and potential vulnerabilities over a specified time frame. Such an analysis is vital for understanding the robustness of the system and

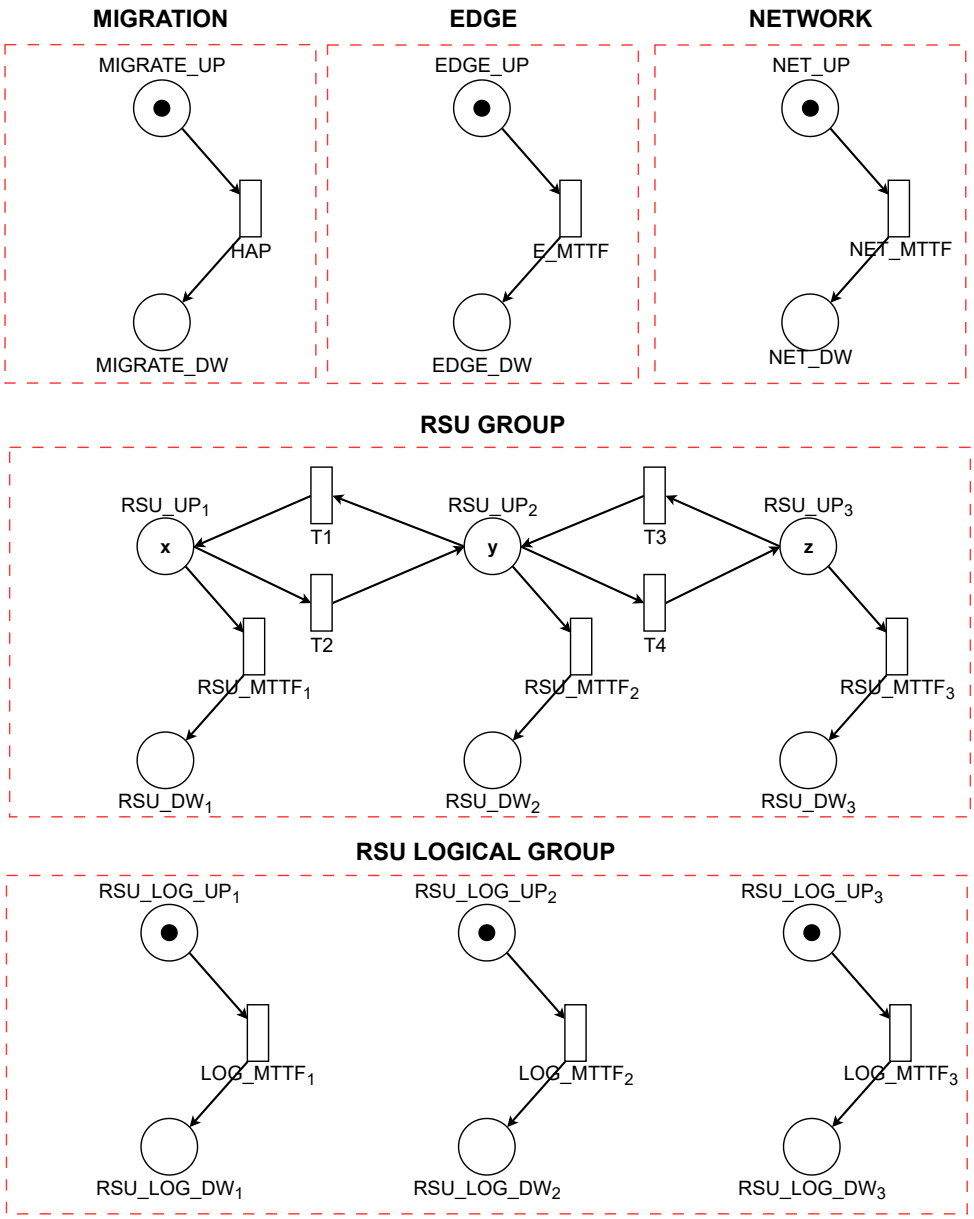


Figura 9 – Base model.

for making informed decisions regarding maintenance, upgrades, or other interventions to enhance the system's reliability.

5.2 Sensitivity Analysis

This research utilizes a DoE approach in conjunction with SPN modeling to derive comprehensive insights into the performance of Edge Computing systems. The variables and their respective levels were consistently applied across models, both incorporating and excluding migration. This methodological consistency was critical to ascertain which variable combinations exerted the most significant impact on the system. By exploring these variable combinations across different scenarios, the study robustly underpins the

optimization strategies proposed for enhancing the system's availability and reliability.

5.2.1 Sensitivity Analysis of the System Incorporating Migration

The sensitivity analysis was conducted using the experimental setup with a primary focus on the time interval preceding system failure. Table 5 presents a comprehensive enumeration of the variables considered in the DoE. This enumeration includes detailed descriptions of each factor and specifies its respective levels. The factors assessed are (a) EDGE_F, (b) NET_F, (c) RSU_F, (d) RSU_R, and (e) LOG_F. Evaluations were conducted at both high and low settings for each factor to ascertain their impact on the overall system performance. The specific configurations for these factors, as applied in the various experimental scenarios, are thoroughly delineated in the table above. The values used were defined based on experimental calibration and controlled variation of the SPN model parameters, allowing the identification of the system's sensitivity to changes in each failure and recovery factor. This empirical approach ensures that the chosen intervals represent operational conditions.

Tabela 5 – Design table

Factor name	**Factor description**	**Low Setting**	**High Setting**
EDGE_F	Edge MTTF	125.892	157.365
NET_F	Network MTTF	83220.0	104025.0
RSU_F	MTTF Physical Part of RSU	500.0	750.0
RSU_R	MTTR Physical Part of RSU	2.0	3.0
LOG_F	MTTF Logical Part of RSU	168.0	210.0

Table comprehensively catalogs the permutations of factor levels employed in the simulations designed to assess their impact on system availability within an Edge Computing framework. Each row in the table represents a distinct amalgamation of factor values, specifically EDGE_F, NET_F, RSU_F, RSU_R, and LOG_F. The terminal column of this table quantifies the system availability corresponding to each unique factor combination. This tabulation effectively encapsulates the outcomes of the DOE simulations, facilitating the identification of factor combinations that either significantly influence or minimally affect the system's availability. The structured presentation of these results serves as a pivotal resource for comprehending the dynamics influencing system performance and the relative importance of various system components.

The graph depicted in Figure 10, illustrating the effects of a DOE with migration, provides pivotal insights into the elements most influential on system availability within an Edge Computing environment. Foremost, the MTTF of the physical RSU emerges as the paramount factor, with its impact value approximating 0.70. This underscores the criticality of physical infrastructure reliability in the overall system performance.

Following this, the MTTR of the RSU's physical components, with values oscillating between 0.40 and 0.45, is identified as another crucial determinant in minimizing system failures. This finding accentuates the significance of efficient repair processes in maintaining system integrity.

The interaction between the MTTR of the physical RSU and the MTTF of the logical RSU is also noted to exert a substantial influence on availability. In addition, the chart delineates factors with comparatively lower impacts, such as the interplay between the MTTF of the Edge component and the MTTF of the Network, the MTTF of the logical components, and the interaction between the MTTF of the Network and the MTTR of the physical RSU, each registering impact values below 0.05. While these elements are deemed less consequential in the present analysis, they could acquire greater significance in certain specific scenarios, suggesting the need for a nuanced understanding of different operational contexts in Edge Computing systems.

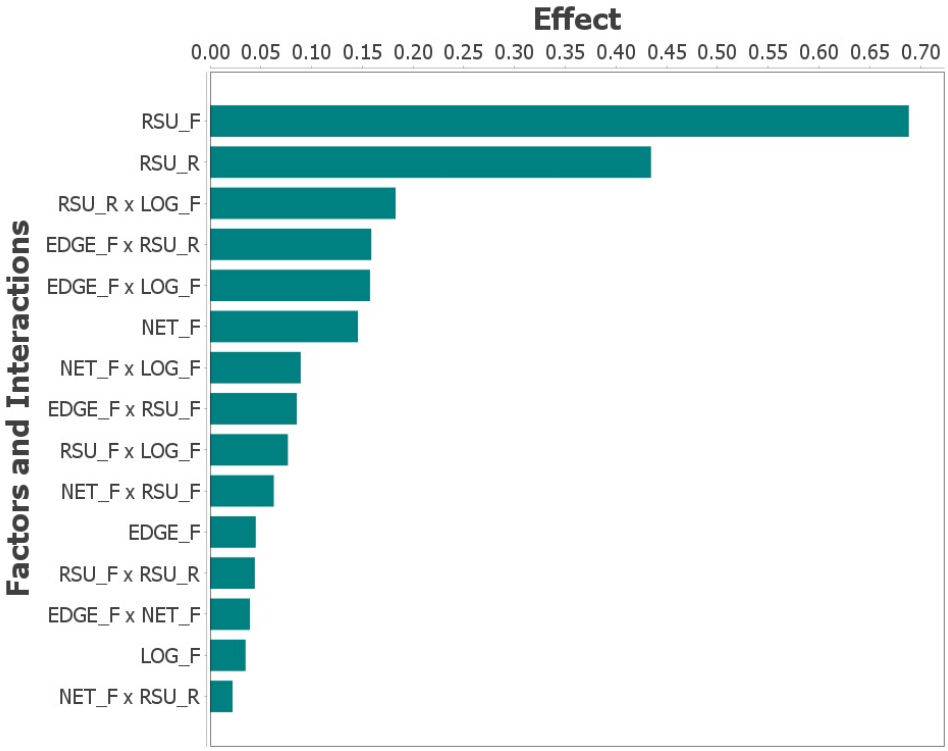


Figura 10 – Impact of different factors on the system with migration

Figure 11 elucidates the intricate interactions between various factors in a migration-inclusive scenario and their collective impact on system availability, as assessed through a DOE approach.

In Subfigure 11(a), the interaction between the Mean MTTF of the Edge component and the MTTF of the Network is analyzed. A notable decrease in system availability is observed when the MTTF of the Edge (157.0) is juxtaposed with the Network's MTTF (98.3). This observation is indicative of the system's heightened sensitivity to fluctuations

in these parameters, underscoring the critical nature of their balance for optimal system performance.

In subfigure 11(b), the dynamics between the MTTF of the Network and the MTTF of the logical RSU component are examined. An inverse relationship is discerned here: an escalation in the MTTF of the Edge (from 168.0 to 210.0) correspondingly diminishes the MTTF of the logical RSU. This pattern exemplifies the complex interdependencies within the system, where enhancing the reliability of one component may inversely affect another, thereby impacting overall system availability.

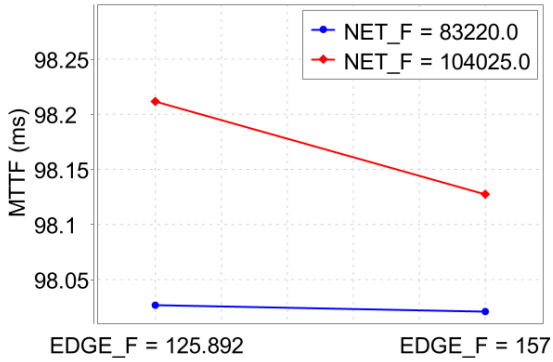
In subfigure 11(c), the analysis is centered on the interplay between the Mean Time to Failure (MTTF) of the Network and the Average Time for Repair (Mean Time to Repair, MTTR) of the physical Road Side Unit (RSU). An observed increase in the Network's MTTF, along with an enhancement to 210 in the MTTF of the physical RSU component, indicates a notable enhancement in overall system performance. This result underscores the critical importance of an integrated assessment of both failure and repair durations within the Network and physical components of the RSU. Such a holistic approach is essential for optimizing system availability. These insights collectively contribute to a deeper understanding of system reliability in Edge Computing environments, highlighting the need for a thorough evaluation of the interactions between various system components to achieve optimal system performance.

5.2.2 Analysis of System Sensitivity in the Absence of Migration

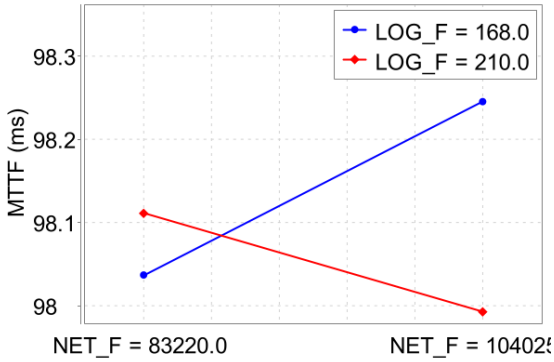
The analysis of the effects obtained by DoE without migration shows that the logical reliability of the RSU (MTTF) is the most determining factor for system availability, with an influence greater than 0.9, indicating its critical importance in the stability of the edge architecture. Next, the MTTR of the physical components of the RSU has a relevant impact (0.6), highlighting the role of repair mechanisms in reducing downtime. Other factors, such as the network MTTF and the interactions between logical and physical components, exhibit smaller, but still significant, effects. Overall, these results show that, in environments without VM migration, system robustness depends mainly on logical reliability and physical recovery capacity, guiding allocation and maintenance decisions aimed at improving availability in Edge Computing architectures.

Figure 13 presents an insightful graph of DoE interactions in a context devoid of migration, offering a clear view of how various factor combinations impact system availability in Edge Computing environments.

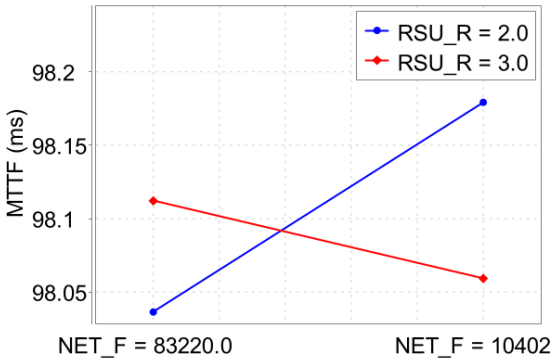
In subfigure 13(a), the graph underscores the system's sensitivity to changes in the MTTF of the Edge component relative to the MTTF of the Network. A significant observation here is that an elevation in the Edge's MTTF to 157.0 prompts a slight increase



(a) MTTF Edge x MTTF Network.



(b) MTTF Network x MTTF RSU Logic.



(c) MTTF Network x MTTR RSU.

Figura 11 – Interaction between factors in the system with migration

in the Network's MTTF, from 95.0 to 95.3. This shift underscores the substantial influence that the Edge's MTTF exerts on overall system availability.

Subfigure 13(b) examines the interplay between the MTTF of the physical RSU and that of its logical counterpart. It is observed that augmenting the reliability of the logical part of the RSU positively influences the MTTF of the physical RSU. This relationship highlights the criticality of integrating these factors to enhance overall system performance.

Lastly, subfigure 13(c) delves into the interaction between the MTTR and the MTTF of the physical RSU. An increase in the MTTR of the physical RSU is shown to improve the MTTF. However, an increase in MTTF does not significantly impact the MTTR. This finding accentuates the importance of a balanced approach to managing failure and repair times, as this balance is key to optimizing system availability.

Together, these insights from Figure 13 emphasize the complex nature of factor interdependencies in Edge Computing systems, particularly in scenarios where migration is not an option. Understanding these relationships is crucial for strategic planning and implementation of measures aimed at enhancing the reliability and efficiency of such systems.

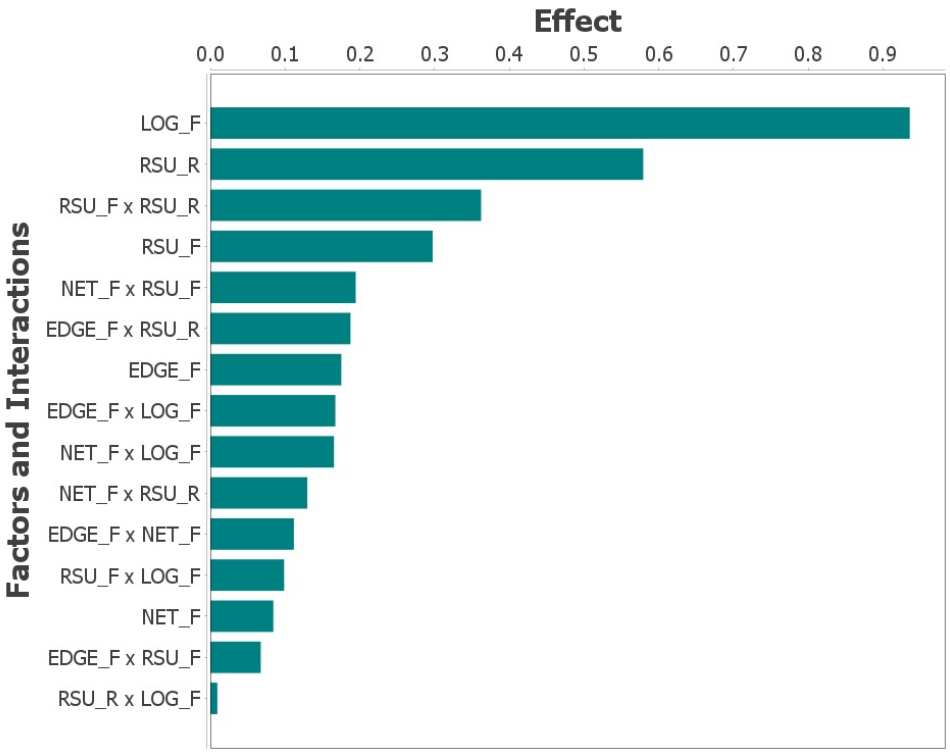
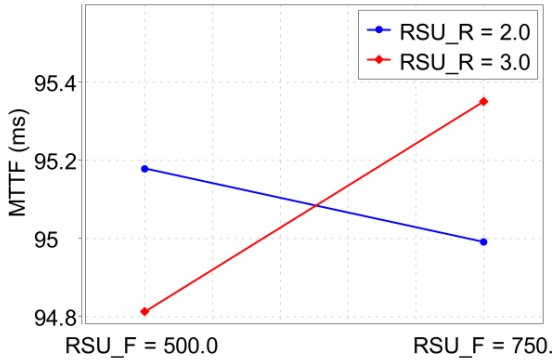


Figura 12 – Impact of different factors on the system without migration

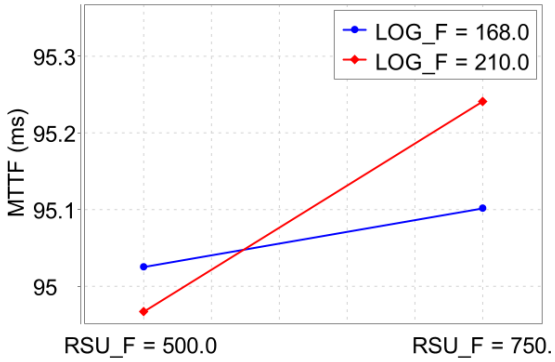
5.3 Case Studies

In this segment, the paper delineates the findings from the analytical evaluation of the models introduced herein. This evaluation encompasses a comprehensive assessment of both the availability and reliability metrics for all the models under consideration. Table 6 provides a detailed account of the parameter values assigned to various system components, with these values meticulously sourced from extant scholarly publications (ARAÚJO et al., 2021b; AUDU et al., 2017; ANDRADE; NOGUEIRA, 2020; MELO et al., 2017). The parameters detailed include those pertaining to the RSU group, the Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR) for the Edge component, and the failure and repair durations associated with the network. Additionally, the table specifies the initial values for the Tokens employed in the system, a critical element in the modeling process.

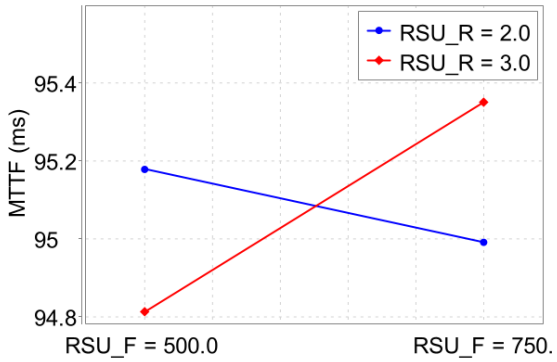
The graph depicting system availability with migration, as shown in Figure 14, delineates a non-linear association between the quantity of VMs utilized and the consequent system availability. Notably, this graph exhibits a convergence of the availability metrics at specific VM counts, namely 8, 16, and 32. These values were obtained empirically from simulation experiments conducted with incremental virtual machine configurations. This overlapping of data lines suggests that the deployment of eight VMs is sufficient to assure the desired level of availability. Such an observation is pivotal in informing strategies for resource allocation and cost optimization. It implies that beyond a threshold of eight VMs,



(a) MTTF Edge x MTTF Network.



(b) MTTF RSU x MTTF RSU Logic.



(c) MTTF RSU x MTTR RSU.

Figura 13 – Interaction between factors in the system without migration

Tabela 6 – Model parameters

Type	Component	Definition	Value
MTTF	EDGE_MTTF	EGDE component failure time	125.89284
	NET_MTTF	Network component failure time	83220.0
	RSU_MTTF	RSU component failure time	500.0
	RSU_LOG_MTTF	RSU Logical component failure time	168.0
MTTR	EDGE_MTTR	EGDE component recovery time	0.913794
	NET_MTTR	Network component recovery time	12.0
	RSU_MTTR	RSU component recovery time	2.0
	RSU_LOG_MTTR	RSU Logical component recovery time	2..0
Variável	TOKENS	Entity representing a state or resource	2.0
	T_M	Migration Time	0.083333

additional VMs do not significantly enhance system availability, thereby offering a pathway to maximize resource efficiency. This efficient allocation of VMs, without compromising the operational efficacy of the system, is essential in balancing cost-effectiveness with system performance.

The research study includes Figure 15, which portrays the graph of system availability in scenarios where migration between RSUs is not implemented. This graph maintains consistency in the range of VMs as used in Figure 14, varying from 2 to 32 VMs in

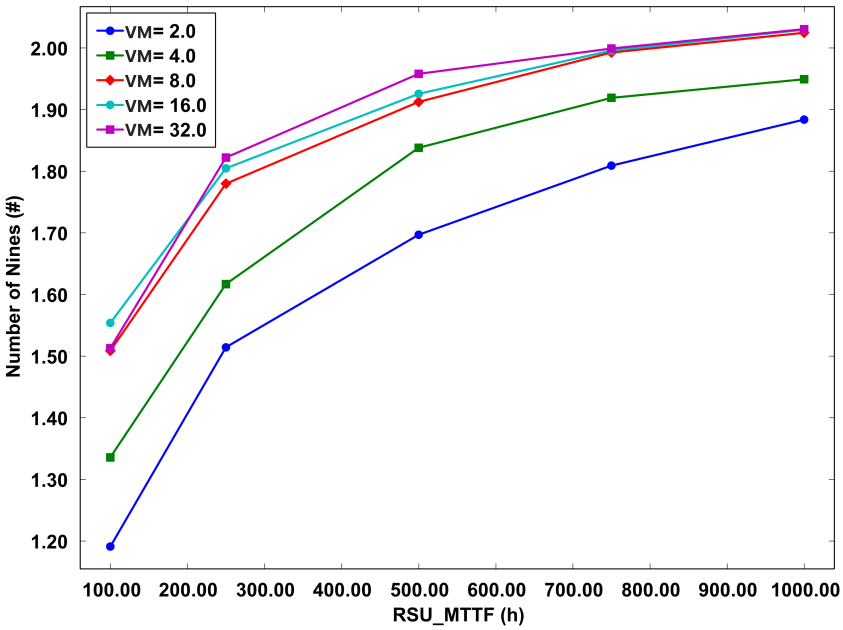


Figura 14 – Availability - Model with migration

operation. This range was defined to capture the progressive scalability behavior of the system, these values were empirically tested during simulation to identify the saturation point where. The MTTF for these VMs is set between 100 and 1000 hours. A critical observation from the results depicted in this graph is the comparative reduction in system availability when migration is not employed, as opposed to scenarios where migration is feasible.

Specifically, the graph shows that system availability commences at '1.00 nines' with the operation of just 2 VMs, and it marginally escalates to '1.36 nines' with the use of four VMs. This pattern of availability underscores the significance of migrating VMs between RSUs to enhance system reliability. The ability to migrate VMs is a crucial factor in achieving higher availability, as evidenced by the increase in the number of 'nines' in the availability metric. This insight highlights the importance of VM migration as a strategy to bolster the robustness and dependability of the system, especially in contexts where maintaining high availability is paramount. The study thus provides a compelling argument for incorporating VM migration between RSUs as a means to optimize system performance and reliability.

In the context of a system deploying four virtual machines (VMs) with an MTTF set at 1000 hours, it is feasible to achieve a level of system availability analogous to that obtained with a deployment of 32 VMs. This finding suggests that an augmentation in the number of VMs does not correspondingly result in a substantial increase in system availability. Particularly in the model excluding migration, the availability attributed to the logical component of the RSUs emerges as a predominant factor, as discerned from

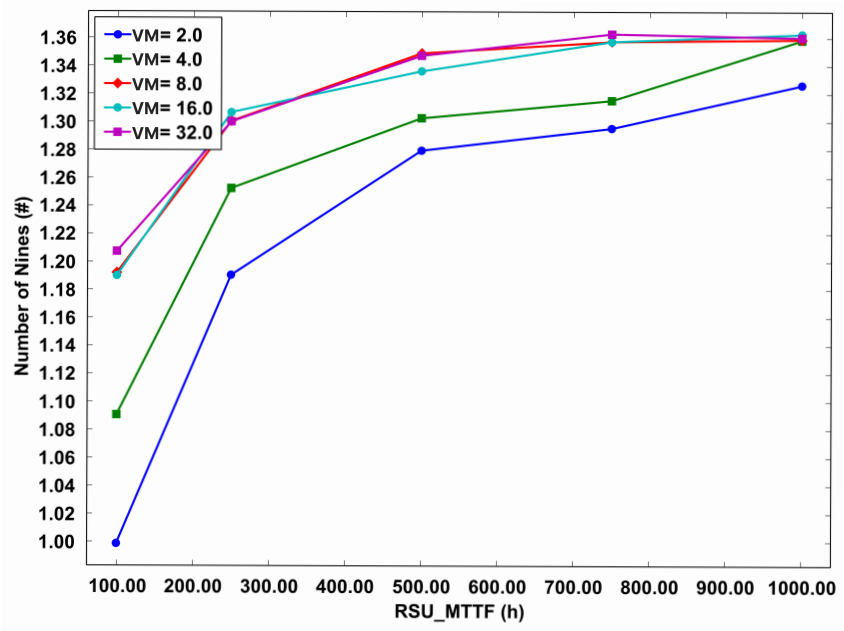


Figura 15 – Availability chart - No migration model

the sensitivity analysis.

This distinction between the models with and without migration underscores the efficacy of the virtual machine migration technique, especially in scenarios characterized by high demand. It accentuates the necessity of integrating VM migration into strategies for computing resource allocation. Figure 16 provides a comparative analysis, juxtaposing the average cases in scenarios of system operation with and without migration. This comparison elucidates the differential impacts of VM migration on system availability, thereby underscoring its critical role in the optimization of computing resources in demanding operational environments.

Figure 16 presents an analysis of system availability based on the failure time of RSUs to assess performance in configurations with and without VM migration. In scenarios without migration, the system demonstrates notable availability, achieving approximately 10.00 nines and increasing slightly to just over 14.00 nines when the MTTF is set at 100 hours. Conversely, in configurations with migration, the availability shows a marked increase as the MTTF is extended, reaching an impressive 20.00 nines with an MTTF of 100 hours. These results clearly indicate that migration exerts a positive influence on system availability, particularly in contexts characterized by extended MTTF periods.

Turning to Figure 17, the reliability graph for the physical component of the system illustrates the variance in the MTTF of the physical aspect of the RSU, with values spanning 168.0, 250.0, and 500.0 over 800 hours. Reliability is a critical metric for evaluating the system's capability to function consistently without failures and interruptions. The data portrayed in this graph establishes a direct correlation between the MTTF of the physical

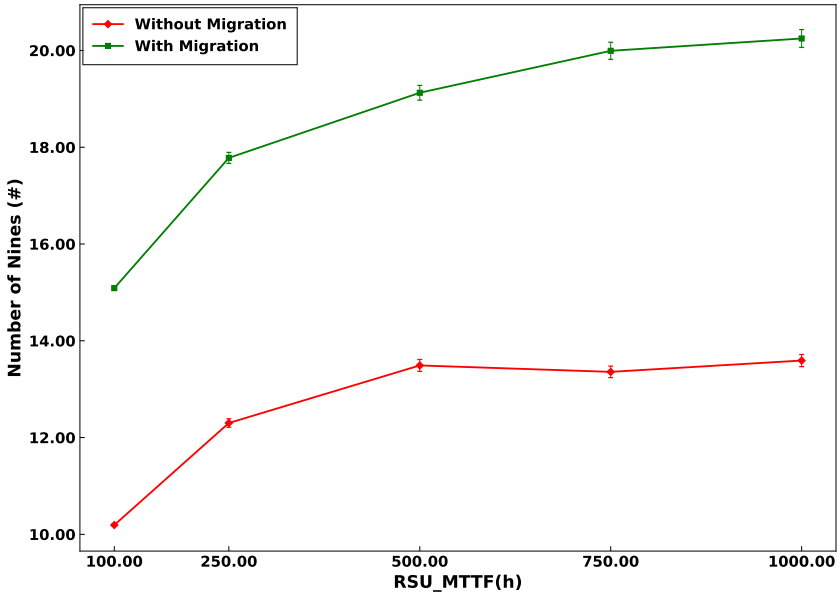


Figura 16 – Comparison between the with migration and without migration models

component and the overall reliability of the system. This relationship indicates that an increase in the MTTF of the physical part is directly proportional to an enhancement in the system's reliability. This insight is pivotal for understanding the impact of the physical component's robustness on the overall operational stability of the system.

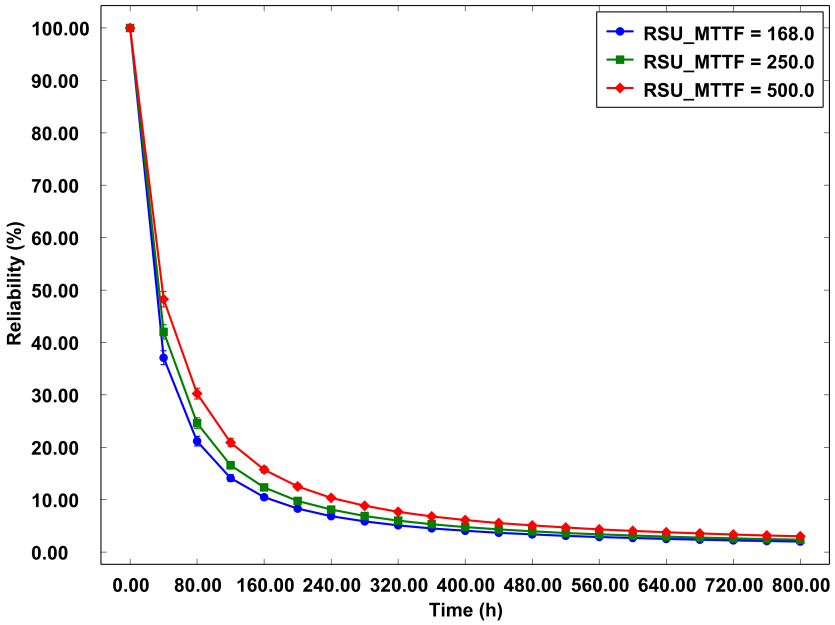


Figura 17 – Reliability - Physical part of the RSU

As the MTTF is extended, the system exhibits enhanced robustness and resilience, thereby diminishing the frequency of failures and bolstering reliable operations. Conversely, a system characterized by an MTTF of less than 168 hours tends to be more vulnerable to failures and faces challenges in sustaining stable operations. This insight is invaluable for

strategizing enhancements to the physical infrastructure of systems in Edge Computing environments. Effective planning in this context encompasses the implementation of both preventive and corrective strategies aimed at augmenting the reliability, quality, and availability of services delivered to end users.

Figure 18 displays the reliability graph for the logical component of the system, with the MTTF varying between 168.0, 250.0, and 500.0 over 800 hours. These values were established to represent different reliability tiers, allowing the observation of how the logical component's resilience evolves across operational lifetimes of virtualized services environments. Assessing the MTTF of the logical part is vital to gauge its capacity to sustain adequate operational performance, particularly in Edge Computing contexts where uninterrupted availability is paramount. The data gleaned from this graph offers insights into the reliability trends of the system's logical component relative to the MTTF of virtual machines.

It is observed that the system with the lowest MTTF of 168 hours exhibits a decline in reliability before reaching 80 hours of operation. This trend signifies a reduced capability of the system to maintain stability and remain free from failures over a shorter duration. In contrast, the system with the highest MTTF of 500 hours demonstrates consistent reliability throughout the initial 80 hours of operation, underscoring its superior ability to remain operational and reliable for an extended period before encountering declines in reliability. These findings are crucial in understanding the resilience of the logical component of the system and in guiding decisions related to the management and optimization of Edge Computing systems.

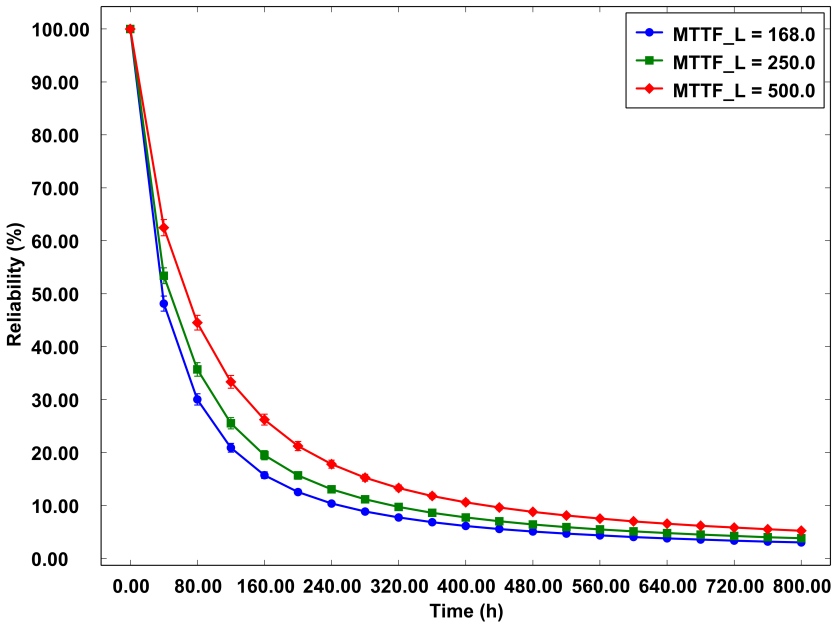


Figura 18 – Reliability - Logical part of the RSU

The data analysis underscores the heightened significance of the logical component in determining the system's reliability, surpassing the influence of the physical part. Although both aspects are integral, the role of virtual machines, particularly their efficiency in terms of MTTF, emerges as a critical factor in ensuring uninterrupted system availability. The observation that a system with an elevated MTTF exhibits prolonged stable reliability prior to any decline in performance indicates that the efficacy and dependability of virtual machines are key determinants of system stability.

Consequently, it is imperative to emphasize strategies aimed at bolstering the reliability of the system's logical aspect. This strategic focus encompasses several key measures:

- Virtual Machine Management Policies: The development and implementation of comprehensive management policies for virtual machines are crucial. These policies should be designed to effectively handle resource allocation, scaling, and migration, thereby enhancing system performance.
- Performance Monitoring: Rigorous and continuous monitoring of system performance is essential. This enables the early identification and resolution of potential issues, thereby maintaining the system's operational integrity.
- Maintenance Practices: The adoption of appropriate and systematic maintenance practices is vital in ensuring the optimal functioning of virtual machines. Regular maintenance activities, including updates and troubleshooting, are necessary for sustaining system health and efficiency.

Implementing these strategies will significantly enhance the availability and quality of the system's services. Such enhancements contribute to the system's resilience and operational efficiency and to the overall user experience in Edge Computing environments. Therefore, prioritizing improvements in the logical part of the system is not merely a technical imperative but a strategic approach to achieving superior service delivery.

6 Performance Analysis

This chapter presents the performance architecture of the proposed system, the stochastic model developed with SPN using the Mercury tool, and the results obtained through sensitivity analyses and controlled case studies. The objective is to quantify the efficiency of VM autoscaling and reinstantiation mechanisms applied to RSUs in dynamic vehicular traffic environments, optimizing the allocation of computational resources based on variable demand.

The architecture includes a road monitoring system capable of horizontally scaling its processing capacity according to the volume of requests received. This scaling occurs through the automatic instantiation of VMs under certain load conditions, while reinstantiation ensures resilience against failures in the VM creation process. The modeled structure incorporates failure detection mechanisms, recovery policies, and removal of idle instances, continuously adjusting resource use to operational conditions. The SPN model was structured in four functional blocks: request admission, waiting queue, processing by RSUs, and control mechanisms (autoscaling and reinstantiation). The transitions were parameterized with stochastic distributions and guard conditions, reflecting the probabilistic and dynamic behavior of the system. Formal metrics were extracted from the model, including mean response time (MRT), utilization (U), discard probability (DP), throughput (TP), and number of VMs in use (NVMU). These metrics allow a quantitative assessment of operational efficiency under different load regimes.

To identify the parameters most sensitive to performance, a DoE was applied, systematically varying factors such as the initial number of VMs, the number of reserved VMs, the failure weight, and the reinstantiation weight. The results showed that the strategic reservation of VMs and the initial system configuration are the main determinants of MRT, being responsible for significant variations in response times even under high traffic. Case studies complement the analysis and demonstrate, through steady-state and transient simulations, the adaptability of the system in scenarios with different traffic intensities and infrastructure failures. The experiments highlight the importance of an efficient balance between elasticity and resilience, allowing the system to maintain its quality of service (QoS) even in stressful situations.

6.1 Model Overview

This section presents the model developed according to the particularities of the

adopted architecture. Figure 19 presents the base model, composed of four parts: (1) Admission, (2) RSU, (3) *autoscaling* Mechanism, and (4) Reinstatement Mechanism.

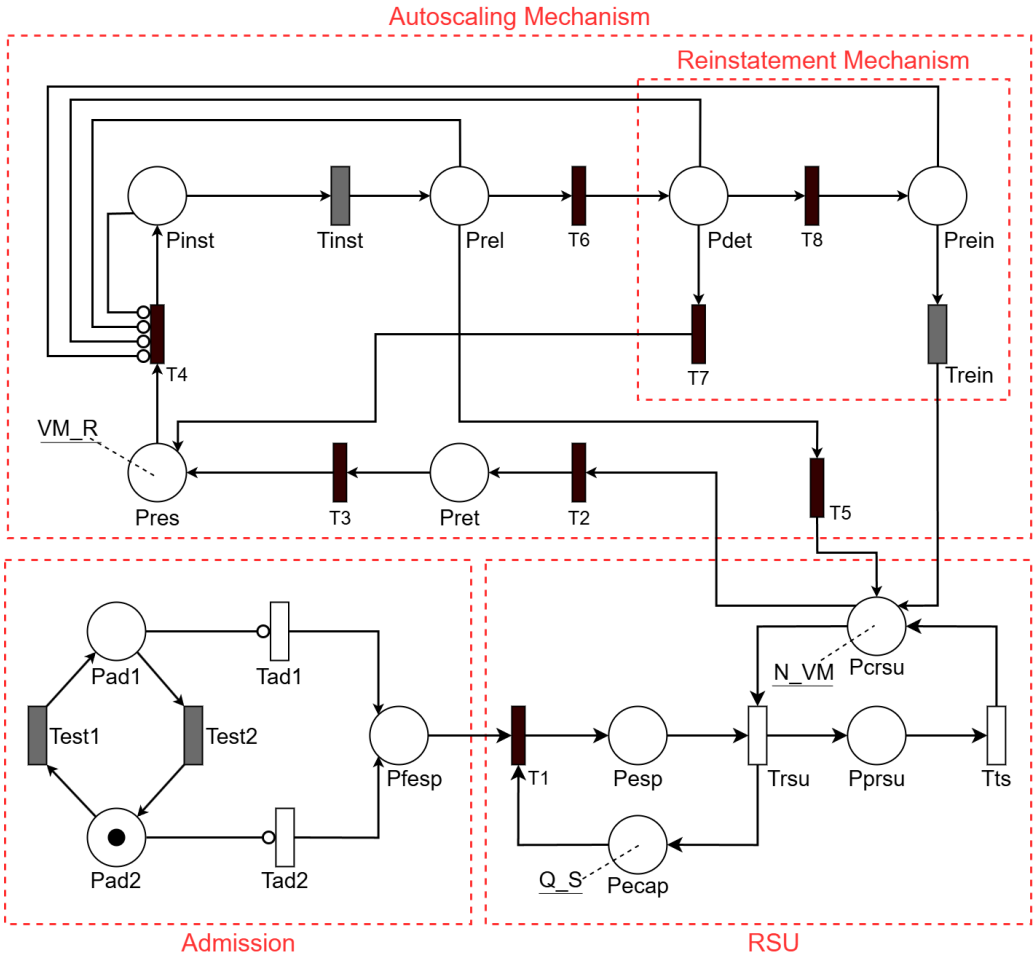


Figura 19 – Model developed based on the adopted architecture.

Admission is the system's first part, and it is responsible for entering requests into the system. It has two admission places (`Pad1` and `Pad2`), which represent different variations in the highway's traffic volume. The two deterministic transitions (`Test1` and `Test2`) indicate the time intervals of change in traffic volume. The place `Pfesp` receives requests generated by the admission places and enables the transition `T1`. The second block of the model consists of the RSU module when the `T1` transition is triggered. The place `Pesp` stores the requests that arrive in the system in a queue, waiting for the data to be passed on for processing.

The `N_VM` represents the initial capacity/quantity of tokens/VMs available in this RSU for processing. The triggering of the transition `Trsu` represents the transfer of the request to start processing the request by the RSU VM, which occurs instead (`Pprsu`). For this processing to be carried out, there must be VMs available in place `Pcrsu`, indicating the RSU's ability to process a request. When the `Tts` transition is activated, the VM processing a request returns to the capacity location.

The third part of the system is the *autoscaling* Mechanism, covering the two stages of autoscaling, which instantiates or frees up system capacity. The first stage is triggered by transition `T4` when it detects an increase in queuing and a reduction in RSU processing capacity (guard condition present in Table 7). In this transition, the reserved ability to instantiate VMs in place `Pres`. Where `VM_R` represents the initial number of virtual machines targeted for autoscaling, it is activated and forwarded to the instance place `Pinst`. Here, the deterministic transition `Tinst` represents the instantiation time of the VM. Which triggers it creates a token in place `Prel`, which directs it to `Pcrsu`, the RSU capacity place, mentioned in the second part of the system, through the transition `T5`. The second stage of automatic autoscaling releases resources when the RSU has low processing demand. When this decrease in demand is identified, the `T2` transition is triggered, collecting the VMs that are no longer in use. These VMs are directed to the fallback place `Pret` and, through the transition `T3`, return to the initial place in `Pres`.

autoscaling can be implemented in two main ways: Vertical and horizontal. Vertical autoscaling increases or decreases a single instance's resources, such as CPU and RAM, making it suitable for directly addressing performance issues. However, it is limited by the physical capacity of the machine and may require downtime. On the other hand, horizontal autoscaling adds or removes instances, distributing the workload across multiple machines. This method offers greater resilience and scalability, allowing dynamic adjustments in the number of instances according to demand, thus ensuring high availability and continuous service efficiency. In this work, we use horizontal autoscaling because it can handle load variations, such as traffic density at peak times. Therefore, horizontal autoscaling is more flexible and efficient, especially in resource-limited environments such as Edge Servers, where workload distribution is crucial for optimized performance.

The fourth part of the system is the Reinstatement Mechanism. Whose objective is to guarantee the successful delivery of the VM to the RSU capacity location in case of failure during instantiation. The `T6` transition is activated, forwarding the VM to the `Pdet` detection Location. In case of failure detection, the VM returns to its original Location through transition `T7`. However, if the failure is corrected, the transition `T8` is triggered, directing the VM to the reinstantiation place `Prein`. Through the deterministic transition `Train`, the VM is directed to the RSU capacity location, `Pcrsu`. This structure seeks to guarantee the system's robustness and operational continuity in the face of potential failures during the instantiation process.

6.1.1 Guard Conditions and Model Metrics

Table 7 presents the implementation of guard conditions in transitions T2, T4, T5, T7, and T8. The guard conditions are necessary to activate the autoscaling and reinstantiation mechanisms. The T2 guard condition is activated when VMs are idle in

`Pcrsu`. It removes idle VMs, returning them to the system reserved VMs place (`Pres`). The T4 condition is activated when `Pcrsu` has less than half of the initial VM capacity, starting autoscaling, in which a new VM will be instantiated. The T5 guard condition refers to the possibility of failure. `P_F` is the failure weight and `P_R` is the reinstatement weight; their respective values are in the Table 7. T7 and T8 are related to the possibility of reinstantiating VMs.

Tabela 7 – Guard conditions used in the model.

Transition	Guard Condition
T2	((#Pcrsu+#Pprsu)(N_VM*(N_VM/2)))AND (#Pprs=(N_VM/(N_VM/2)))
T4	(#Pcrsu=(N_VM/(N_VM/2)))
T5	(1-P_F)
T7	(1-P_R)
T8	(P_R)

For this work, metrics for Average Response Time, Use, Probability of Disposal, and Flow were obtained. Measurement of the Number of VMs Used (NVMU) is calculated by adding the expected number of capacity and in-use tokens present in `Pcrsu` and `Pprsu`, as presented in Equation (6.1).

$$\mathbf{NVMU} = (E\{\#Pprsu\}) + (E\{\#Pcrsu\}) \tag{6.1}$$

Equation (6.2) presents the system's average response time. The average response time (MRT) can be obtained from Little's Law (JAIN, 1990a). This law requires a stable system, that is, one that has a request rate lower than the server processing rate. The law indicates that the Average Response Time (MRT), in our case, is called the Mean Response Time (MRT). It is given by dividing the average number of requests in progress in a system ($RequestsInProgress$) by the arrival rate of new deliveries (AR). The arrival rate is the inverse of the interarrival time. We abbreviate the time between arrivals here to AD. The value of $RequestsInProgress$ is the sum of tokens in each Location representing a request in progress. $Eplacename$ represents the statistical expectation of there being tokens in "placename", where $Eplacename = (\sum_{i=1}^{n} P(m(Local) = i) \times i)$, being n the largest number of tokens than the $Local$ may contain. In other words, $Eplacename$ indicates the expected value of tokens at that Location. This metric is the sum of the expected number of requests present in the queue and being processed in the RSU. Which is taken into account and multiplied by the time between new request arrivals. To follow Little's Law, it is necessary to "discount"the discarding of requests in the system, dividing the MRT by the probability of tokens not being discarded.

$$\text{MRT} = \frac{E\{\#Ppqe\} + E\{\#Pprsu\}}{\left(\frac{E\{\#Pprsu\}}{\text{Ttsrsu}}\right)} \tag{6.2}$$

The resource utilization level (U) is calculated by adding the resources used and dividing by the total previously available capacity. The use of the system is given by Equation (6.3). The utilization is multiplied by 100, as Mercury gives the probabilistic value of the model between 0 and 1.

$$\mathbf{U} = \frac{E\{\#Pprsu\}}{N\_VM} * 100 \tag{6.3}$$

The Equation (6.4) represents the discard probability (DP) of the system. DP reflects the chance of a request being rejected when the system reaches its maximum capacity. In this context, DP was used to understand the system's ability to deal with variations in demand. Discarding occurs when there are no more resources available to meet a new request. The place responsible for monitoring this information is (`Pecap`), which checks the availability of resources simply by observing whether it is null. If the value of (`Pecap`) is null, this indicates that the system has reached the limit of its capacity, resulting in a rejection of new demands. The Equation (6.5) presents the system flow rate. Flow becomes relevant in this context, as its result reflects the system's ability to scale horizontally to meet variable traffic demand. It is measured by the expectation of tokens at the RSU processing location, divided by the time it takes for the subsequent request to be processed (Tts).

$$\mathbf{DP} = P\{\#Pecap = 0\} * 100 \tag{6.4}$$

$$\mathbf{TP} = \frac{E\{\#Pprsu\}}{Tts} \tag{6.5}$$

6.1.2 Assessment of Structural and Behavioral Properties

This section presents an analysis of the structural and behavioral properties of the developed model, according to the particularities of the adopted architecture, as discussed in the book (MACIEL; LINS; CUNHA, 1996). The evaluation of structural properties, such as structural limitation, conservation, repeatability, and consistency, is fundamental to ensure that the system operates within defined limits. Likewise, it is essential to analyze behavioral properties, including reachability, boundedness, security, liveness, coverage, persistence, reversibility, and fairness. This ensures that the system responds appropriately to requests, avoids idle or failure states, and handles all requests equitably. This section details these properties, highlighting their implementation in the model and their relevance to the system's operational efficiency.

6.1.2.1 Structural Properties

The structural limitation of the model is evidenced by the explicit definition of the initial number of VMs (tokens) available in the RSU for processing, as represented by the location `N_VM`. This fixed capacity of tokens/VMs ensures that the system cannot exceed a predefined number of processing resources, preventing system overload and maintaining operation within established limits. Conservation is guaranteed by the resource release and reuse mechanism, where VMs return to their initial location (`Pres`) after being deactivated. This reuse cycle ensures that resources remain constant in the system, neither being created nor destroyed, but being redistributed as needed. As stated, the VMs are directed to the replacement location `Pret` and, through the transition `T3`, return to the initial location in `Pres`.

Repeatability is incorporated into the model through the reinstatement mechanism, which allows failed VMs to return to their original state and be reintegrated into the system for reprocessing. This process ensures that the system can return to previous states repeatedly while maintaining operational continuity. The goal is to ensure the successful delivery of VMs to the RSU capacity location in case of failure during instantiation. System consistency is maintained by the ability to dynamically adjust resources according to demand. Autoscaling, which instantiates or releases VMs as needed, ensures that the system maintains a constant operational balance by adapting to workload fluctuations. As mentioned, the second stage of autoscaling frees up resources when the RSU has low processing demand.

6.1.2.2 Behavioral Properties

Admission is the first part of the system and is responsible for entering requests into the system. Reachability is demonstrated by requests' ability to traverse multiple parts of the system, from intake to processing and reinstatement. This ensures that different system states and operations are accessible and achievable as needed. Limitation is guaranteed by the requirement that VMs must be available at the `Pcrsu` location for processing to occur. This control prevents the system from processing more requests than its capacity allows, avoiding overcrowding and ensuring operation within defined limits. For processing to be carried out, VMs must be available at the `Pcrsu` location, indicating the RSU's ability to process a request. In the model, security is observed through the strict control of the number of VMs available for processing, preventing the simultaneous presence of multiple tokens in locations that would represent an unsafe condition. This control prevents operational failures and maintains system integrity. The `N_VM` represents the initial capacity, the number of tokens, and the VMs available in this RSU for processing.

The system's coverage is evidenced by its ability to adjust processing capacity according to detected demand. This dynamic adjustment allows the system to cover

different operational scenarios, responding appropriately to changes in workload. When a decrease in demand is identified, the `T2` transition is triggered, collecting the VMs that are no longer in use. Finally, system persistence is ensured by storing requests in a queue until they can be processed. This ensures that requests do not lose their eligibility for processing and remain in the system until they are adequately met. When the `T1` transition is triggered, the `Pesp` location stores the arriving requests in a queue.

6.1.3 Model Limitations

In this section, we will discuss the limitations of the developed model. The assumption was made that all system activities occur in exponential time. However, it's important to highlight that, being a simulation, other time distributions can be used to represent the system dynamics more realistically. The choice of an exponential distribution was based on its widespread application in simulation models, particularly for modeling events that occur independently and continuously.

The variations introduced into the model, such as sudden changes in the time distributions used, demonstrate how the system can respond to different traffic scenarios. Significant fluctuations in traffic rates are common in real-world environments, characterized by periods of low and high demand. During peak times, the arrival rate of requests can increase dramatically, whereas during times of lower vehicle density, this rate can decrease considerably. These variations present challenges to the system's autoscaling mechanism, which dynamically adjusts processing capacity to maintain efficiency and quality of service.

The use of time distributions other than exponential can enhance simulation robustness, providing a more realistic assessment of system performance across various scenarios. Therefore, the model's limitation is that it assumes exponential times for all processes, which may not fully capture the complexity of real-world situations. Simulation offers significant advantages, including cost reduction and resource savings, while also allowing exploration of various variables and conditions to optimize system performance.

6.1.4 Sensitivity Analysis

This study applies advanced DoE techniques to conduct a rigorous sensitivity analysis, allowing for the quantitative evaluation of the impact of variables on system behavior (WEISSMAN; ANDERSON, 2015). This structured approach involves planning and executing controlled experiments, modifying input variables to understand their influence on the metric of interest, the MRT. The analysis employs interaction and effects plots to identify dependencies between factors and prioritize those with the greatest impact on system response, facilitating optimization and resource allocation decisions (LENDREM; OWEN; GODBERT, 2001). Factor levels were defined according to industry standards

and empirically adjusted in preliminary simulations—with parameters such as the number of virtual machines (N_VM = 4–8) and the queue size (QS = 100–1000)—ensuring representativeness of typical operating conditions in edge computing environments.

The factors adopted for this study are represented by five system components: (i) *N_VM* (ii) *VM_RES* (iii) *Weight_Reins* (iv) *Weight_Fail* (v) *QS*. The factors have two levels, which are low and high settings. Table 8 shows the levels executing the DoE. Table 15 displays the results of executions related to the combinations used to create the DoE.

Tabela 8 – Design of experiments.

Factor name	**Low Setting**	**High Setting**
N_VM	4.0	8.0
VM_RES	4.0	48.0
Weight_Reins	0.1	0.3
Weight_Fail	0.1	0.3
QS	100.0	1000.0

6.1.5 Results Analysis

This subsection presents the results of the sensitivity analysis with DoE. Figure 20 presents the factor effect graph. This effect is the weight that each factor exerts individually on the system. The VM_RES factor, the number of reserved VMs, demonstrated the most significant impact on the system's average response time. The second factor with the most significant impact was the interaction between the number of initial VMs in the system (N_VM) and VM_RES, followed by N_VM. The VM_RES factor had an impact greater than 25,000. The interaction between VM_RES and N_VM impacted approximately 25,000, as did just the N_VM factor. When comparing the other interactions and factors, they had a low impact on the MRT. Such data allows the system to improve the MRT, as the components/factors with the most significant impact on the system have been identified.

Figure 21 graphs DOE interactions in the context of autoscaling, offering a clear view of how various combinations of factors impact the system's MRT. Figure 21a highlights the system's sensitivity to changes in the number of initial VMs regarding the queue size. A significant observation here is that increasing the number of initial VMs causes the average response time to decrease. Notably, the queue is less congested as processing capacity has increased. This change underscores the substantial influence that the Number of Initial VMs has on overall system performance.

Figure 21b examines the interaction between the Number of Initial VMs and the Number of Reserved VMs. Regarding MRT, when the number of reserved VMs is 48.0, the number of initial VMs does not change when the number is increased. Under these conditions, the number of reserved VMs meets the system's needs. When the number of

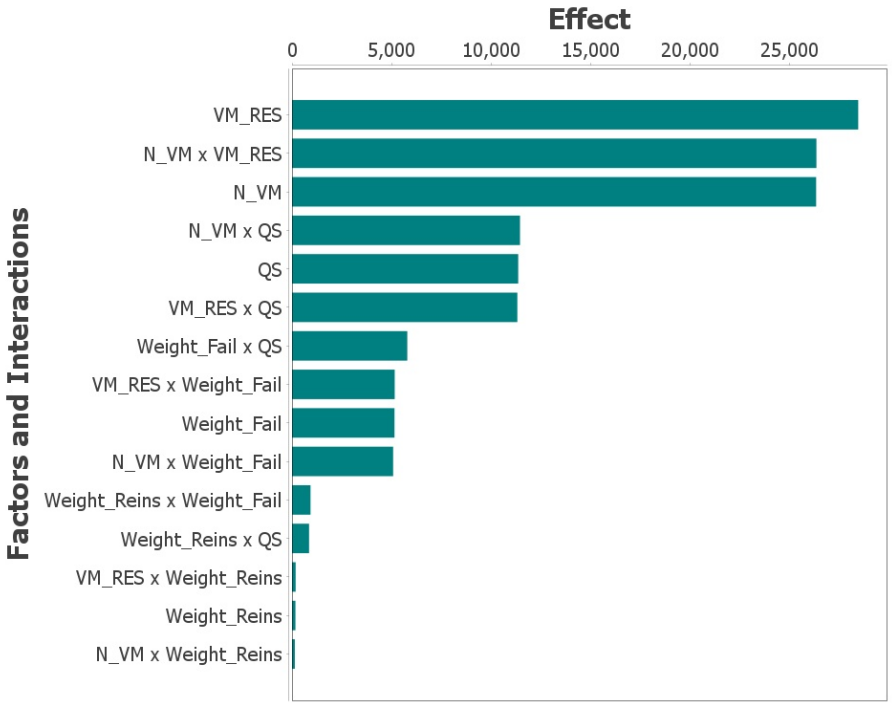
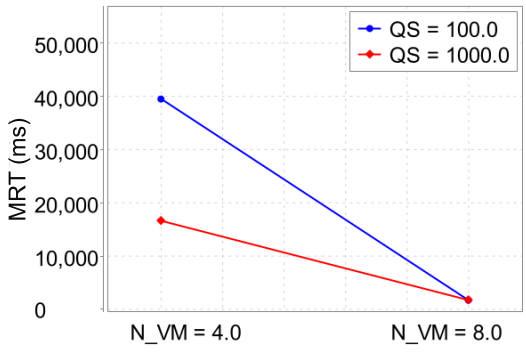
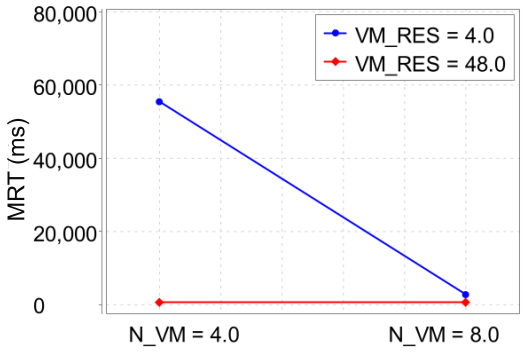


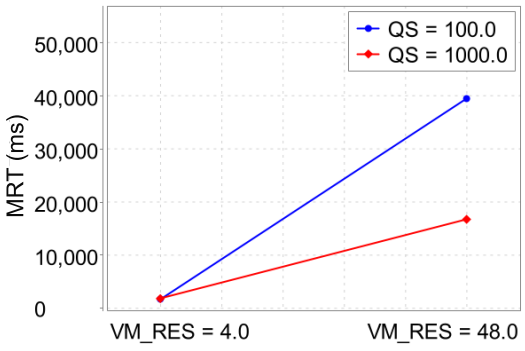
Figura 20 – Visualization of the impact of various model factors on the MRT metric.



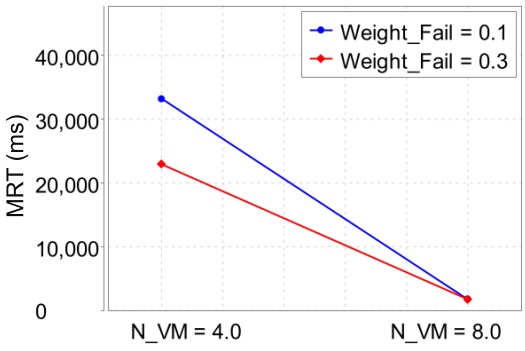
(a) Num. of initial VMs x Queue Size



(b) Num. of initial VMs x Num. of reserved VMs



(c) Num. of VMs reserved x Queue Size



(d) Num. of initial VMs x Fail Weight.

Figura 21 – Interaction between factors regarding their impact on the MRT metric

reserved VMs is only 4.0, increasing the initial VMs from 4.0 to 8.0 directly influences the MRT.

Figure 21c investigates the interaction between the Number of reserved VMs and Queue Size. It is shown that increasing the number of reserved VMs improves the MRT compared to the queue size. When the queue is 100.0, its MRT is approximately 40,000. When the queue is 1000.0, its MRT is between 10,000 and 20,000. This finding highlights the importance of a balanced approach, as this balance is critical to optimizing system performance.

Figure 21d checks the interaction between the Number of initial VMs and the Failure Weight. It is shown that increasing the number of initial VMs improves the MRT, even if the failure weight also increases. These data emphasize the complexity of factor interactions in autoscaling systems, especially in scenarios where system performance is indispensable.

6.2 Case Studies

This section presents the results obtained as case studies. Due to the complexity of the model's nature, the case studies were carried out using stationary simulations. An approximate margin of error of 2% was incorporated. The parameters used in the configuration of the simulations of the model in Figure 19 are presented in Table 9.

The values assigned to the parameters of the system components were derived from previously published academic sources, especially in the works (ANDRADE; NOGUEIRA, 2020; MELO et al., 2017). Transitions timed are configured as single server, except the `Tts` transition, which is configured as an infinite server. The following subsections highlight the potential use of the model.

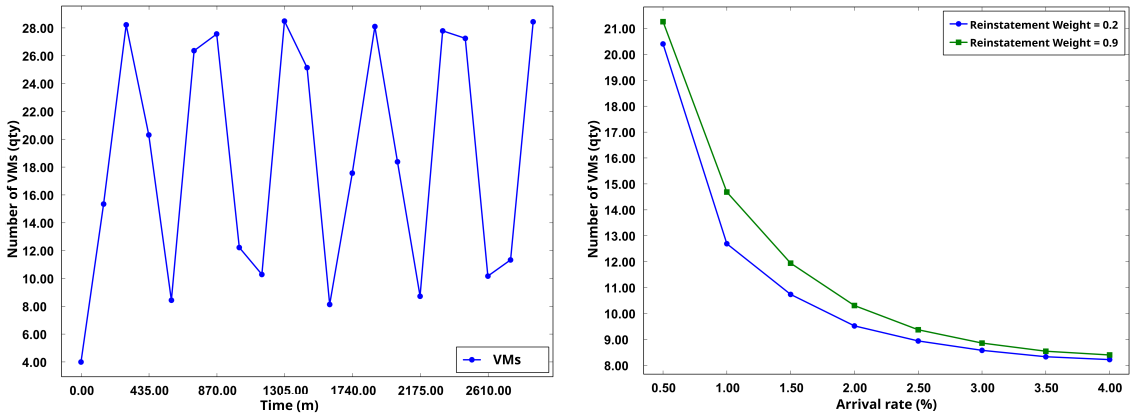
6.2.1 Case Study 1 - System Adaptability Analysis

This subsection presents the adaptability results obtained through the system analysis. One of the central points of the work was to ensure the efficient availability of the system during traffic peaks at different times of the day. This, combined with the prevention of wasted resources through the autoscaling mechanism, includes reducing idle VMs at night, for example. Furthermore, to save resources, the VM reinstantiation rate was measured, seeking to ensure that the instantiation process was repeated if forwarding a VM to the RSU failed. Figure 22 presents both the use of adaptability through the variation of VM demand throughout the day and the instantiation process in case of failure.

Figure 22a shows the variation in the use of VMs throughout the day (metric NVMU). Its adaptability depends on the state exchange used in the model, initially with

Tabela 9 – Parameters used in the model.

Type	Component	Value	Definition
Places	Pad1, Pad2	1.0, 0.0	New requests in the system.
	Pfesp	0.0	Request queue.
	Pesp	0.0	Queue of requests to process.
	Pecap	Q_S	Request queue capacity.
	Pprsu	0.0	RSU processing.
	Pcrsu	N_VM	RSU capacity.
	Pres	VM_R	VMs reserved in the system.
	Pret	0.0	VMs return to place.
	Pinst	0.0	VMs instantiation place.
	Prel	0.0	VMs release place.
	Pdet	0.0	Failure detection place.
	Prein	0.0	Reinstantiation place.
Transitions Timed	Test1, Test2	100.0	Transition between systems.
	Tad1, Tad2	AD1, AD2	Time interval.
	Trsu	0.1	Request arrival time.
	Tts	15.7	Arrival transition at RSU.
	Tinst	1.0	Service time transition.
	Trein	2.0	VM instantiation transition.
			VM reinstantiation transition.
Variables	AD1, AD2	10.0, 0.7	Arrival of data.
	Q_S	1000.0	Queue Size.
	N_VM	4.0	Number of RSU initial VMs.
	VM_R	25.0	System reserved VMs.
	P_F, P_R	0.1, 0.1	Failure Weight and Reinstatement Weight.



(a) Number of VMs in Use (NVMU). (b) Reinstantiating VMs on the System.

Figura 22 – System adaptability to variation in the number of VMs.

4 VMs. For the analysis to be viable, time was defined in minutes. This period totaled 2,880 minutes, equivalent to 2 days, which is the time present on the x-axis. In the base analysis, there were 25 reserved VMs available, plus the initial 4 in the RSU. During the first period, all available reserved VMs are used. After that, when car traffic on the highway decreases, the system detects that the number of VMs is no longer needed. This

avoids wasting resources and allows us to forward these VMs to another RSU. Thus, the graph line shows a decreasing behavior.

A Transient Analysis method was used to make this analysis possible. This analytical method offers a detailed and dynamic view of the system's temporal evolution, providing essential information about its ability to adapt to fluctuations in demand over time. Due to its dynamicity, transient behaviors, such as load peaks, can be detected. This was possible through variable states of the model (Test1, Test2 from Table 9).

Figure 22b represents the results of the system reinstantiation mechanism. This mechanism sought to reduce the failure when instantiating new virtual machines for the RSU. From the increase in the arrival rate percentage (x-axis of the graph), the y-axis represents the number of reserved VMs. This analysis started with a number above 21.0 Virtual Machines. Notably, VMs are instantiated when the system is first used, and the arrival rate is still lower than 1.0

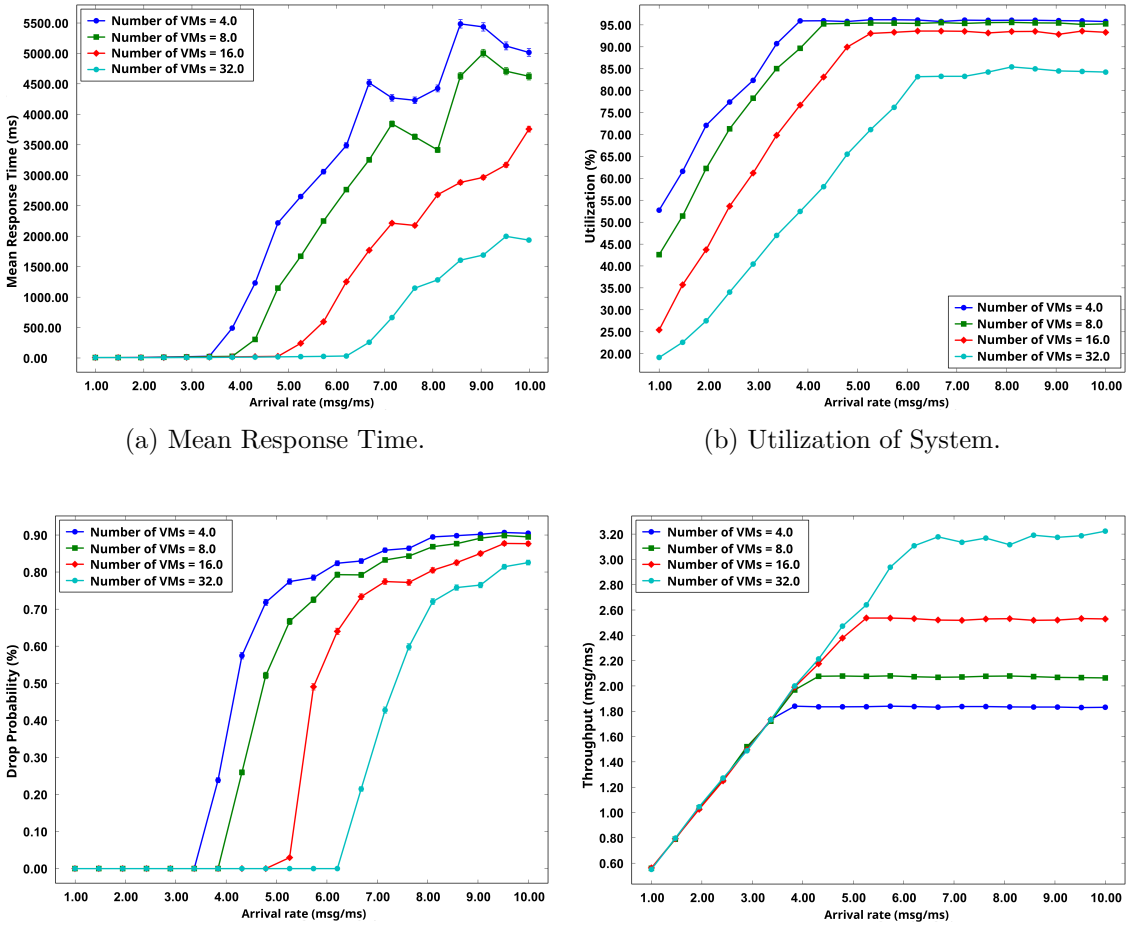
As the arrival rate percentage increases, the number of VMs shows a decreasing behavior; that is, VMs are being routed to the RSU through autoscaling. This demonstrates that the reinstatement mechanism's behavior is as expected since there is a continuous drop reaction. If the instantiation mechanism fails, the lines should increase behavior, as the reserved VMs would return to their origin.

6.2.2 Case Study 2 - Performance Analysis

This subsection analyzes the performance results of varying the initial number of VMs in the system. Throughout the analysis, the number of VMs emerged as a preponderant factor in the system's response time and effective use. Graphs corresponding to the disposal probability and system flow are also incorporated into this evaluation to provide a more comprehensive understanding of the results.

Figure 23 presents the analysis results that address the variation in the initial number of VMs. This figure is related to the performance of typical VANET behaviors when simulating the computational capacity distributed among vehicles connected to the network. This variation is related to the average response time (MRT), system usage, drop probability, and system throughput (TP). Figure 23a shows a significant increase in Mean Response Time as the arrival rate increases. It is essential to highlight that reducing the initial number of VMs results in a greater MRT in the system. For example, when using 4 VMs (the lowest initial value used), the system maintains satisfactory performance up to 4.00 messages per millisecond. From which the MRT increases progressively, reaching approximately 5,500.00 ms. The other results indicate an increasing trend. However, when the number of initial VMs is increased, this increase is not as pronounced as in the case of the lowest initial value used. The highest value, for example, 32 initial virtual machines,

starts to grow from approximately 6.00 messages, with the MRT reaching approximately 2000.00 ms.



(a) Mean Response Time.

(b) Utilization of System.

(c) Drop Probability.

(d) Throughput.

Figura 23 – Impact of varying the number of virtual machines on system performance

Figure 23b shows the percentage of system utilization based on the variation in the initial number of VMs. It can be seen that when the system starts, it uses more than 50% with 4 VMs; it reaches its maximum peak of 95% even before 4.00 msg/ms. The second variation with 8 VMs follows, and it also reaches maximum system usage, above 95%, however, from 4.00 msg/ms. The most significant number of VMs used in the system did not reach the maximum peak, remaining approximately 85% of usage from 6.00 msg/ms. High usage, as in the first two variations, leads to a higher discard rate, considering that the system is stressed.

Figure 23c demonstrates the system's drop probability. The overall results show that the system is stressed to a maximum of 90% with 4.0 VMs and 8.0 VMs. More than 89% with 16.0 VMs, and approximately 88% with 32 VMs. The discard graph presented here plays a complementary role to MRT. Analyzing the relationship between the two, we observed that the MRT begins to grow, later reaching an inflection point where it

begins to decrease. This behavior suggests an interesting dynamic in the system, where the discarding of messages is inversely related; as MRT grows, the probability of discarding influences the system's operational efficiency.

The flow, represented in Figure 23d, is a function of the arrival rate for different numbers of Virtual Machines (VMs). This illustrates the system's ability to process messages in a given time interval. With 4 VMs, the flow reaches its peak, approximately 1.80 to 2.0 msg/ms on the y-axis, before stabilizing around 4.00 msg/ms on the x-axis. On the other hand, through autoscaling, for example, when using 32 VMs. The throughput continuously grows up to approximately 3.20 msg/ms on the y-axis and stabilizes at 6.0 msg/ms on the x-axis. Therefore, the throughput reflects the system's efficiency in handling the workload, revealing the appropriate adjustment of the number of virtual machines.

6.2.3 Case Study 3 - Initial VMs x Reserved VMs

This section presents the result of varying the Number of initial VMs available ready to be used along with the Number of VMs reserved for system response time. Figure 24 shows the result of varying the number of initial VMs and the number of Reserved VMs for the system's average response time. For this experiment, a range of the number of initial VMs was defined between 4 and 24 VMs. The number of reserved VMs was defined for the experiment between 8 and 48.

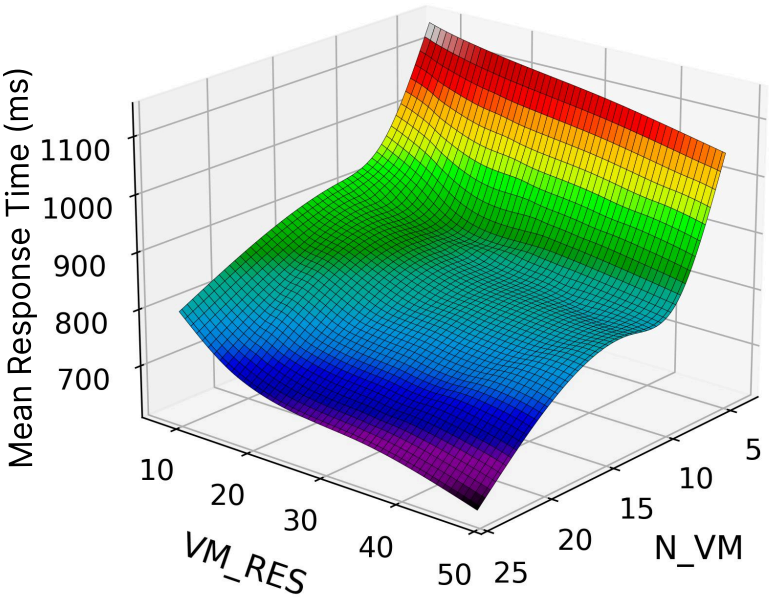


Figura 24 – Simultaneous variation in the number of initial VMs with Number of Reserved VMs.

The graph displays the reduction in MRT as the number of reserved VMs (VM_RES) in the system and the initial number of VMs (N_VM) increase. When the maximum number of VMs is available (in this analysis, VM_ RES=48 and N_VM=8), the average

response time is below 700 ms. When capacities start to be reduced, the MRT tends to increase. This is due to the decrease in system capacity, impacting its performance. In the worst-case scenario, where VM_RES=8 and N_VM=4, the MRT has its maximum peak, exceeding 1100 ms. Therefore, increasing only the initial number of VMs is sometimes the most efficient choice, as the MRT tends to remain high with few VMs reserved.

This analysis demonstrates the need and balance provided by autoscaling in an environment with constant variation. Therefore, when increasing the intensity of monitored traffic, it is crucial to exercise caution to add enough new VMs for the system to perform well and avoid wasting resources. In the tests carried out for this graph, the growth of the MRT is not excessive. However, it indicates that autoscaling guarantees system performance and the necessary quantities to avoid wasting resources. The results also corroborate previous DoE findings, which show that the number of initial VMs and the number of reserved VMs are the most impactful factors on system capacity. This behavior is observed in VANET scenarios, where vehicular networks face congestion and have their efficiency compromised. This analysis highlights the importance of self-scaling in the network, dynamically adjusting capacity to balance vehicular traffic and avoid overload and resource waste, which is typical behavior of a VANET in dynamic environments.

7 Sustainability Analysis

This chapter presents the proposed sustainability studies and the model validation. The proposed SPN model for sustainability analysis in vehicular infrastructure preserves the same conceptual structure as the previous performance model, with modifications. The proposal evaluates the energy efficiency of the system under different operating conditions, considering the impact of autoscaling and reinstantiation policies on energy consumption and carbon emissions. The validation was performed with experimental data obtained using the Pasid-Validator tool, ensuring consistency between the simulation and the behavior observed in the real world.

Initially, related works addressing load balancing and energy efficiency in VANETs are discussed, identifying methodological gaps and highlighting the importance of formal models to capture the stochastic behavior of these systems. The inclusion of these complementary studies was necessary to support the selection of metrics associated with energy, efficiency, and sustainability, expanding the analytical scope of the proposed model. Subsequently, these works served as a reference for defining the factors evaluated in the sensitivity analysis. The model maintains the Admission, Waste Management System (WMS), Automatic Scheduling, and Reinstantiation modules, with new metrics associated with energy consumption. The variables Energy Consumption (EC), Energy Efficiency (EE), and Carbon Emission Factor (CEF) have been incorporated.

This configuration allows for the evaluation of the energy cost of each operation, correlating performance and environmental impact. While the performance model measures time and flow rate, the sustainability model quantifies the energy consumed and associated emissions, expanding the scope of the analysis to aspects of efficiency and environmental responsibility. The calibration of the parameters was based on the average arrival, processing, and reinstantiation times obtained experimentally. The system behavior was analyzed under load and capacity variations, allowing observation of how automatic scheduling influences energy consumption and the utilization level of the processing units. The results confirm that dynamic scheduling policies reduce energy waste without compromising system stability.

Finally, sensitivity analysis assessed the influence of key factors on energy consumption and overall efficiency. The most relevant variables were the number of VMs (N_C), reserved capacity (C_R), and failure and reinstantiation weights (P_F, P_R). The case studies demonstrated that optimized autoscaling configurations reduce total energy consumption and carbon footprint while maintaining performance within acceptable limits. The model confirms the viability of sustainable federated architectures, balancing elasticity,

performance, and environmental impact.

7.1 Methodological Approach

This section presents the methodological approach adopted in this study, structured to ensure clarity and reproducibility of the proposed model. First, we describe the overall methodological process, illustrated by a flowchart, which outlines the sequential stages of the research — from literature review and architectural definition to modeling, validation, sensitivity analysis, and case studies. Next, we detail the system architecture, which represents the foundation of the Stochastic Petri Net (SPN) model and guides the evaluation of autoscaling and sustainability strategies in vehicular networks.

7.1.1 Considered Architecture

This section describes the architecture developed to create the SPN model. Figure 25 illustrates the proposed architecture to integrate dynamic scaling and energy sustainability in vehicular environments with RSUs. The system is designed to operate adaptively, responding to changes in vehicular traffic volume over time. The central component is the RSU unit, which has elastic capacity to meet different levels of load generated by connected vehicles. As the flow of vehicles increases, message generation intensifies, resulting in a greater number of requests that need to be processed by the RSU. To avoid overload and ensure quality of service, the system incorporates an autoscaling mechanism that allocates additional VMs as demand grows. Allocation occurs in an automated and reactive manner, ensuring that the infrastructure dynamically follows traffic demands, without resorting to permanent oversizing.

The architecture is based on the principle that the volume of traffic on highways varies significantly throughout the day, directly influencing the rate at which vehicles generate data. During periods of low traffic, the system operates with a reduced number of VMs, consuming only the resources necessary to maintain service responsiveness. However, during peak times, the infrastructure needs to respond quickly to increased computational load, avoiding bottlenecks that compromise V2I communication performance. To this end, the autoscaling mechanism is activated reactively whenever the occupancy of the request queue exceeds a certain threshold. This threshold, set at 80% of the queue capacity, was defined based on empirical tests and represents a critical inflection point that precedes saturation. This preventive policy allows new VMs to be instantiated before the system reaches an overloaded state, maintaining a balance between performance and energy consumption.

The process of allocating and deallocating VMs occurs from two logical repositories: The idle VM pool and the set of active VMs in operation. When the autoscaling policy

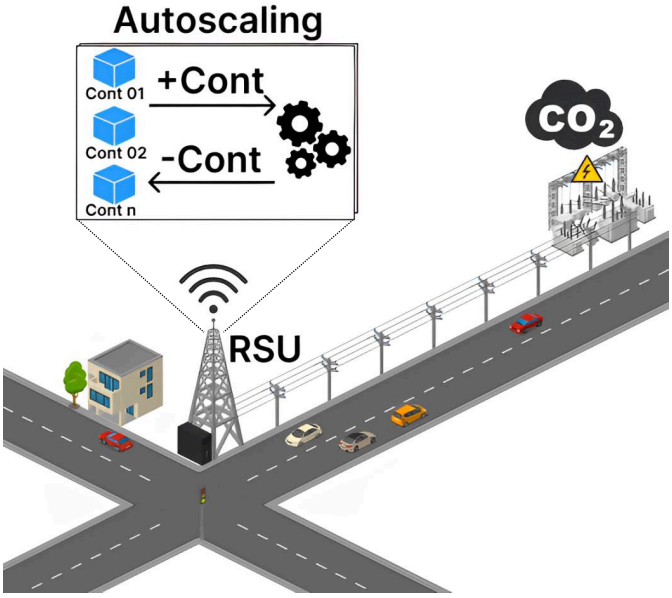


Figura 25 – Proposed architecture for adaptive RSUs with autoscaling and energy-aware operation in VANETs. The system reacts dynamically to variations in traffic volume, adjusting the number of active VMs to maintain performance and minimize resource waste.

identifies the need for expansion, new VMs are instantiated from the pool and added to the RSU processing capacity. Similarly, when the request load decreases and underutilized instances are identified, the system controls the removal of excess VMs and returns them to the pool. This strategy prevents computational resources from remaining unnecessarily active, which is essential to reduce energy consumption without compromising service continuity. The scaling logic is also sensitive to failures during the instantiation process. To address this, the reinstantiation module is used, which automatically detects and corrects failures, ensuring infrastructure resilience.

7.1.2 Overall Methodology

The methodological process adopted in this study was structured in sequential stages, ensuring consistency between conceptual foundations, formal modeling, empirical validation, and case-based application. Figure 26 illustrates the main phases followed throughout the research.

- **Literature Review:** The first stage consisted of a systematic review of works addressing autoscaling, energy efficiency, and performance optimization in VANETs. This step allowed identifying the main strategies employed in the state of the art, as well as their limitations regarding sustainability metrics. The review also highlighted the potential of Stochastic Petri Nets (SPNs) as a formal tool for modeling adaptive systems, guiding the definition of the present proposal.

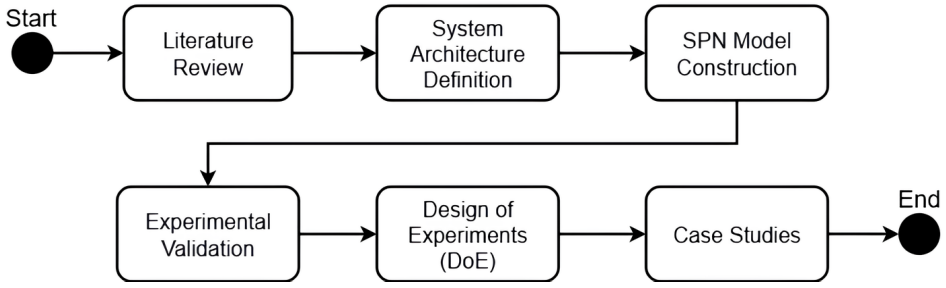


Figura 26 – Overall methodological framework adopted in this study, comprising the sequential stages: literature review, system architecture definition, SPN modeling, experimental validation, DoE-based sensitivity analysis, and case studies.

- **System Architecture Definition:** Based on the insights obtained from the literature, an adaptive RSU architecture was specified, integrating dynamic autoscaling and reinstantiation mechanisms. This architecture was conceived to reflect real vehicular environments, where computational demand fluctuates with traffic intensity and where failures must be efficiently handled to maintain service continuity. The architecture served as a blueprint for the SPN model.

- **SPN Model Construction:** The proposed architecture was formally represented through SPNs, enabling a probabilistic analysis of system behavior. The model was structured into four modules: Admission, RSU Processing, Autoscaling, and Reinstantiation. Each module was designed to capture the stochastic dynamics of request arrivals, resource allocation, fault detection, and recovery, thus providing a detailed yet analytically tractable representation of the system.

- **Experimental Validation:** To ensure that the proposed model accurately reflected real operational dynamics, an empirical validation process was conducted. Using the Pasid-Validator tool, execution times were collected in a controlled environment, instrumented with a request generator and a load balancer node. These data were used to parameterize the SPN transitions, allowing simulation results to be statistically compared with experimental Mean Response Times (MRTs). This stage confirmed the representativeness and fidelity of the model.

- **Design of Experiments (DoE):** Once validated, the model was employed in a sensitivity analysis using DoE. This methodology provided a structured way to evaluate the influence of critical factors — such as the number of VMs, reserved capacity, and failure/reinstantiation probabilities — on energy consumption and efficiency. By applying statistical techniques, DoE enabled quantifying the relative impact of each factor and their interactions, offering deeper insights into the trade-offs between performance and sustainability.

- **Case Studies:** Finally, two case studies were designed to demonstrate the applicability of the model. The first compared system behavior with autoscaling enabled and disabled, highlighting the energy and environmental gains of adaptive mechanisms. The second analyzed the effect of varying the number of VMs, evidencing the trade-off between scalability and energy efficiency. These case studies illustrate the practical relevance of the model for system planning and optimization in VANET scenarios.

This methodological sequence ensured that the study progressed in a rigorous and coherent manner: from the identification of theoretical gaps, through the construction of a formal and validated model, to its application in scenarios that demonstrate both technical performance and sustainability implications.

7.2 Model

This Section details the developed model, structured in four main modules, which represent the system operations: (1) Admission, (2) RSU Unit, (3) Autoscaling System, and (4) Reinstantiation System. These modules were designed to analyze the system's performance in relation to traffic dynamics and computational demand, while respecting the parameters of the adopted architecture.

7.2.1 Model Structure

Figure 27 illustrates the main structure of the model. The Admission module controls the initial flow of requests. In it, place `Tad` represents the incoming requests generated by traffic, while place `Pqe` organizes the transfer of requests to the next stage. The RSU module then processes pending requests. Queued requests are stored in place `Ppqe` until sufficient resources are available in place `Pcaprsu`, indicating VMs ready for execution. The transition `Trsu` transfers the request to place `Pprsu`, where processing occurs. After processing, the transition `Ttsrsu` returns the VMs to place `Pcaprsu`, allowing their use in new operations.

The autoscaling module dynamically manages the system's capacity based on demand. When a high queue of requests and capacity reduction is detected, transition `T4` is activated, triggering the instantiation of new VMs from location `Pcres`. These VMs pass through location `Pic`, where the time required for instantiation is simulated by transition `Tic`, before being sent to location `Pcaprsu` via transition `T5`. Conversely, when demand decreases, idle VMs are released by transition `T2`, temporarily stored in `Pretc`, and reallocated to the initial reservoir `Pcres` via transition `T3`.

Finally, the Reinstantiation module handles failures during the instantiation process, ensuring the system operates continuously. When a failure occurs, transition `T6` redirects

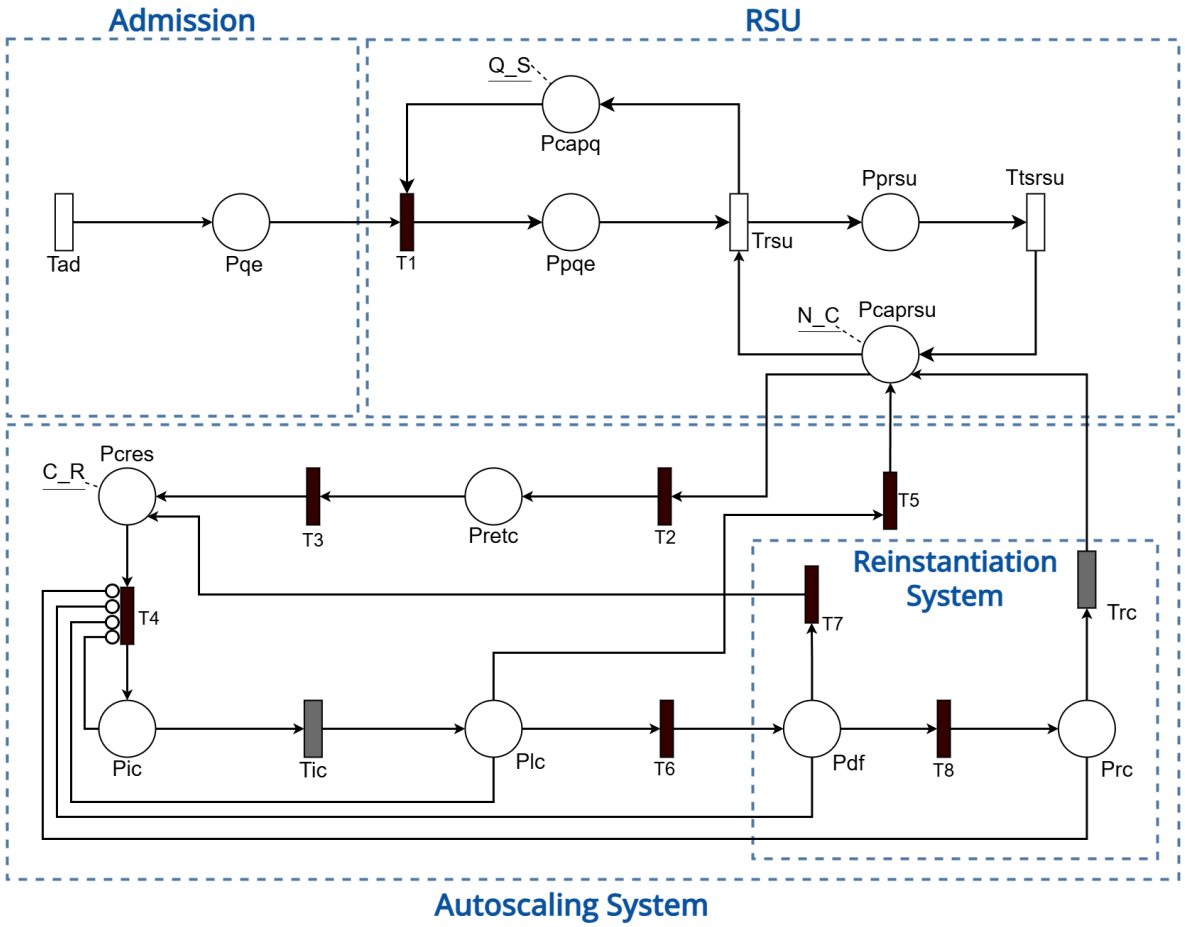


Figura 27 – SPN model representing the dynamic behavior of RSUs. The model comprises four modules: Admission, RSU Processing, Autoscaling, and Reinstantiation, interconnected to simulate real-time response to traffic load fluctuations and failures.

the VM to the detection location `Pdf`, which tries to correct the failure. If successful, the VM proceeds to the `T8` transition. If the failure is not resolved, the `VM` returns to the initial reservoir `Pcres` via transition `T7`. If the failure is corrected, transition `T8` forwards the VM to location `Prc`, where the deterministic transition `Trc` finalizes the process by reallocating the VM to the capacity location `Pcaprsu`. This approach strengthens the resilience and operational efficiency of the system. The model employs exponential distributions for all timed transitions, consistent with the classical semantics of SPNs. This choice ensures analytical tractability and allows rigorous derivation of steady-state measures. Although phase-type or general distributions could provide a more detailed representation of bursty vehicular traffic, the exponential abstraction was adopted to balance realism with mathematical solvability.

7.2.2 Model Metrics and Guard Conditions

In the context of this work, the metrics are focused on evaluating the energy impact of the system. The metrics used in the experiments were Energy Consumption (EC), Carbon Footprint (CF), and Energy Efficiency (EE). Each of these metrics is essential to measuring performance in terms of resource consumption and environmental impact, considering the system's behavior under different loads and operating conditions.

Equation 7.1 contains the EC metric, which aims to calculate the total energy used by the active components of the system during operation. To do this, the calculation takes into account the energy consumed by both the RSU and the virtualization VM (Cont), considering the total operating time. The calculation is based on the system utilization, checking the expectation of having tokens in `Pprsu` and `Pcaprsu`, that is, requests being processed by the RSU. `E_OP_RSU` is the energy required to keep the RSU active, and `E_OP_Cont` is the energy required to keep the VM running. The factor `Time` represents the duration of time that these components are active and processing information. The calculation considers the total energy consumed per hour by the components while they are operating simultaneously.

$$\mathbf{EC} = ((E\{\#Pprsu\} \times E\_OP\_RSU + E\{\#Pcaprsu\} \times E\_OP\_CONT) \times Time) \quad (7.1)$$

The metric that calculates the CF, contained in Equation 7.2, calculates the environmental impact of the system's energy consumption, considering the carbon emission factor (CEF). This factor depends on the energy source used, since different energy sources (such as coal, gas, or renewable energy) have different Carbon Dioxide (CO2) emission intensities. The difference between this metric and Energy Consumed is that the Carbon Footprint takes into account the environmental impact of the energy used, multiplying the total energy consumed by the CEF. The CEF is a constant that reflects the amount of CO2 emitted per unit of energy consumed, and is essential for verifying greenhouse gas emissions. This makes the Carbon Footprint a metric that assesses the sustainability of the system in terms of environmental impact. It should be emphasized that the adopted energy consumption model assumes linear proportionality between the number of active tokens (requests/VMs) and the energy used. This abstraction was chosen to simplify the stochastic representation, which may play a relevant role in real-world RSU deployments.

$$\mathbf{CF} = CEF \times EC \quad (7.2)$$

Equation 7.3 covers EE and is based on the relationship between completed tasks, represented by the number of successfully processed tasks, and the energy consumed. This calculation seeks to reflect the efficiency of the system in using its energy resources to

perform operations. Based on the system throughput, the metric is a proportion between the requests being processed, the service time, and the energy consumption. It is measured by the expectation of tokens in the place of RSU processing `Pprsu`, divided by the time it takes for the subsequent request to be processed `Ttsrsu`, after which it is divided by the energy consumption. Thus, it reflects the system's ability to perform its tasks using the least amount of energy possible, and is essential for evaluating the system's performance in terms of sustainability and resource optimization.

$$\mathbf{EE} = \left(\left(\frac{Pprsu}{Ttsrsu} \right) \div EC \right) \times 100 \tag{7.3}$$

The Mean Response Time (MRT) of the system is presented in Equation 7.4. Usually, the MRT can be obtained from Little's Law (JAIN, 1990b), which requires system stability. That is, the request arrival rate must be lower than the processing rate. However, in this work, we adopt a metric based directly on the expected quantities of tokens (requests) observed in the stochastic model. The equation expresses the MRT as the ratio between the sum of the expected quantity of requests waiting in the RSU queue (`E{#Ppqe}`) and in processing in the RSU (`E{#Pprsu}`). This total is divided by the effective average service rate of the RSU. This rate is represented by the ratio between the expected number of requests being processed in the RSU (`E{#Pprsu}`) and the average service time of the RSU (`Ttsrsu}`). In this context, `E{#Ppqe}` and `E{#Pprsu}` represent, respectively, the statistical expectation of tokens in the places that model requests waiting in the queue and processing. The denominator of the equation provides the RSU with an effective throughput. This value is calculated based on the RSU processor's average occupancy and the time required to complete each service. Thus, the MRT reflects the average time required for a request to travel through the system. This estimate considers both the queue dynamics and the service capacity of the RSU. The adopted metric is more appropriate to the stochastic model used, since it directly incorporates the expected values observed in the system.

$$\text{MRT} = \frac{E\{\#Ppqe\} + E\{\#Pprsu\}}{\left(\frac{E\{\#Pprsu\}}{\text{Ttsrsu}} \right)} \tag{7.4}$$

Table 10 details the guard conditions associated with transitions T2, T4, T5, T7, and T8, which are fundamental for the dynamic control of the system. These conditions are necessary for the activation of the autoscaling and reinstantiation mechanisms of VMs. For transition T2, the guard condition is triggered when the number of idle VMs in `Pcaprsu` is detected. In this case, the inactive VMs are moved back to the reserved VMs storage location, `Pcres`, optimizing the available resources. The T4 transition is triggered when the number of VMs in `Pcaprsu` represents less than half of the initial capacity, which causes the autoscaling mechanism to be activated. However, the `SWT` variable that defines the

activation must be equal to 1, thus activating autoscaling. This results in the instantiation of a new VM to meet the growing demand of the system.

Tabela 10 – Guard conditions used in the model.

Transition	**Guard Condition**
T2	((#Pcaprsu + #Pprsu) > (N_C*(N_C/2))) AND (#Pprsu <= (N_C/(N_C/2)))
T4	(SWT = 1) AND (#Pcaprsu <= (N_C/(N_C/2)))
T5	P_F
T7	1 - P_R
T8	P_R

In the proposed model, P_F (failure weight) and P_R (reinstantiation weight) are routing weights assigned to the immediate transitions T5, T7, and T8, encoding probabilistic branching without inflating the state space. In transition T5, the guard condition refers to the detection of system failures: when a failure is identified, the transition is activated and the token follows the failure path with weight P_F or the success path with 1-P_F, allowing the system to execute the necessary procedures to maintain uninterrupted operation. Transitions T7 and T8 are directly linked to the reinstantiation process: T7 is triggered when a VM must be reinstantiated after a failure, while T8 is activated when the reinstantiation succeeds and the VM is restored to its functional state. By concentrating heterogeneous causes of failure and recovery (e.g., transient contention, software restarts, or short-lived network issues) into compact weights, the model preserves analytical tractability while explicitly capturing the autoscaling and reinstantiation logic.

7.3 Model Validation

This section describes the validation process of the proposed SPN model to represent the behavior of VM autoscaling systems in VANETs. The validation seeks to verify the accuracy of the model by reproducing the real behavior of the system under different load conditions, obtained through controlled empirical experimentation. The methodological procedure was structured in stages, including the construction of the model, the instrumentation of the experiment, the collection of data, and the comparison between the simulated and experimental results. The main metric used for analysis was the MRT, which reflects the average time required for a request to travel through the system. The following presents the adopted methodology and the details. After that, the experiment's architecture, the validation tool, the comparative results, and the final discussion are presented.

7.3.1 Validation Methodology

The validation methodology adopted in this work was structured in sequential steps, as illustrated in Figure 28. The process begins with system modeling, moves on to instrumentation of the experimental environment, and ends with statistical analysis of the adherence between model and experiment. Each step is described in detail below.

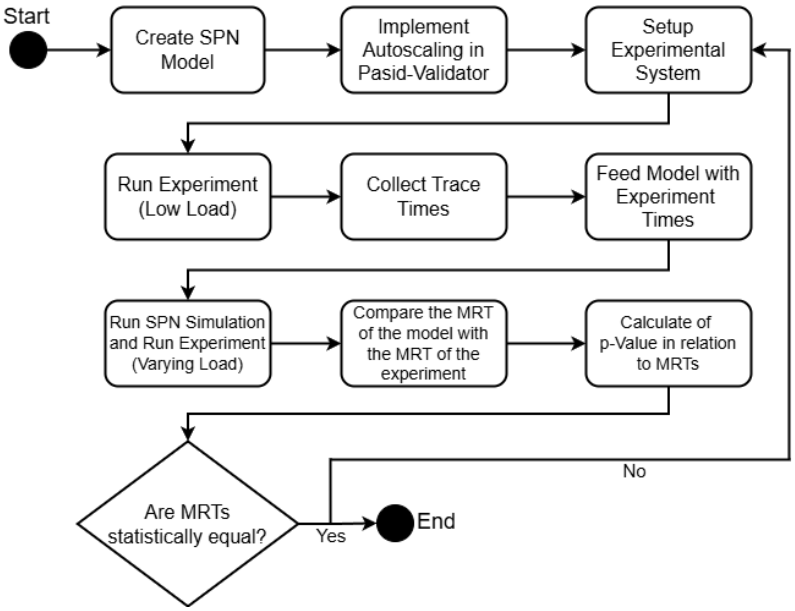


Figura 28 – Validation methodology for the proposed SPN model. The process includes model construction, instrumentation of the experimental environment using Pasid-Validator, empirical data collection, simulation, and statistical comparison of MRT values.

- **Create SPN model:** The model presented in Section 5, based on Stochastic Petri Nets, was used to represent the autoscaling system under continuous request arrival. The model structure includes queues, multiple parallel instances, and a reactive scaling policy. For validation purposes, the model parameters were adjusted to reflect the real system configurations.
- **Implementation of the autoscaling logic in Pasid-Validator:** The scaling policy was coded directly in the load balancing component of the Pasid-Validator tool[1]. The added logic monitors the queue occupancy and dynamically instantiates new services whenever the usage exceeds the 80% threshold.
- **Setup Experimental System:** The physical environment consisted of two main nodes: a source node, responsible for request generation, and a load balancer node, responsible for dynamic distribution and automatic scaling control. The model was structured to represent the specified architecture.

[1] <https://github.com/LuisGuilherme-x7/Pasid_Validator_Autoscaling>

- **Execution of the experiment with low load:** Initially, the system was executed with a reduced request rate, in order to allow the stable collection of intermediate processing, transmission, and response times.
- **Collection and mapping of intermediate times:** During the execution with low load, Pasid-Validator recorded the times between relevant events (such as sending, receiving, and completion of the request). These average values were used to feed the model transitions directly.
- **Model feeding with experimental times:** The collected times were incorporated into the SPN model, ensuring that the simulation considered the real dynamics observed in the physical environment.
- **Model simulation with different loads:** The model was run in steady state with multiple arrival rate configurations. For each case, the MRT and standard deviation were obtained.
- **Practical execution with load variation:** In parallel to the simulation, the real system was run under the same arrival rate configurations. For each scenario, the experimental MRTs were collected from the Source component instrumentation.
- **Comparison between model and experiment:** The results obtained in the simulations and the physical experiments were directly compared. The adherence was verified based on the proximity between the MRT values obtained in the two domains.
- **Calculation of p-Value in relation to MRTs:** From the data obtained from the simulation and the experiment, the p-Value between the MRTs in each scenario is calculated.
- **Final statistical validation:** The compatibility between the results was statistically evaluated. The model was considered validated whenever the simulated MRT remained within the 95% confidence interval of the experimental observations.

7.3.2 Experiment Architecture and Model Used

The experimental architecture was designed to reproduce the main elements of the modeled system. The main focus was on the service autoscaling logic and the vehicle infrastructure based on RSUs. The physical environment consisted of two machines interconnected in a network, as illustrated in Figure 29. The Source is responsible for generating requests and simulating the demand from vehicles in the VANET. This machine emulates the behavior of mobile nodes, sending requests continuously according to the configured arrival rate parameter. The LoadBalancerProxy represents the RSU, being

responsible for the queue and for dynamically distributing requests among active services. This machine implemented the autoscaling logic, which monitors the occupancy of the request queue and instantiates new services reactively.

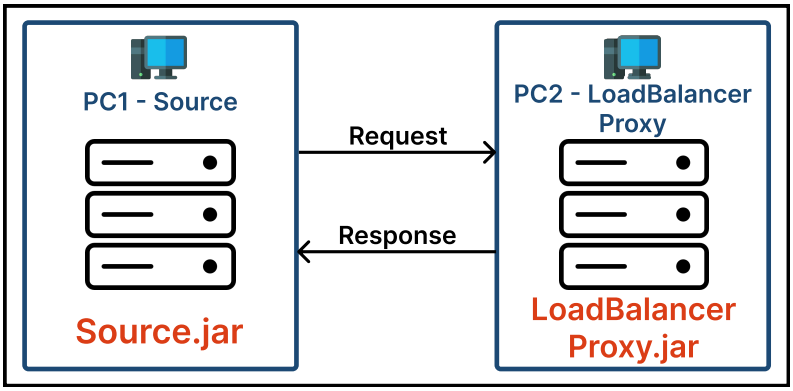


Figura 29 – Experimental system architecture. Two networked machines were used: PC1 (Source) generates vehicular requests, while PC2 (Load Balancer) processes them using dynamic autoscaling policies in response to queue occupancy thresholds.

The scheduling policy adopted in the experiment follows the same criteria as the SPN model. Whenever the utilization of the request queue exceeds 80% (COHERENCE, 2023) of CPU usage, a new service instance is automatically created. This value was chosen because it represents a saturation point close to the limit of the queue capacity, allowing the growth in demand to be anticipated without incurring excessive delays or loss of performance. Each instance is represented by an independent thread, capable of processing requests in parallel. The number of active services, therefore, varies dynamically according to the intensity of the load. The infrastructure was implemented with the help of the Pasid-Validator tool, extended with support for adaptive scheduling. This extension allows the same adaptation mechanisms represented in the model to be replicated in the physical environment, ensuring structural and behavioral coherence between the experiment and the simulation.

For validation, the SPN model presented in Figure 27 (Section 7.2) was used. To ensure equivalence with the experimental environment, the model parameterization values were adjusted to faithfully reflect the average service times, queue size, and arrival interval adopted in the real experiment. The automatic instantiation logic was also kept identical to that used in the instrumented code of Pasid-Validator, ensuring that both the model and the experiment respond equivalently to demand variations. The model was executed in stationary mode, with multiple simulations for each AR value. In each case, the MRT was calculated considering the sum of the times in the queues and in the active servers, according to the classic definition of Little's Law, demonstrated in Equation 7.4. The experimental setup was intentionally designed with two physical machines to provide a controlled and reproducible environment for validation. This setup allowed for precise

instrumentation of vehicular networks with request arrivals, queue occupancy, and service times, ensuring statistical comparability with the SPN model. With this configuration, the model was able to simulate the adaptive behavior of the system, allowing a direct comparison with the results observed in the practical experiment, as will be discussed below.

7.3.3 Pasid-Validator

The model validation was conducted using the Pasid-Validator tool. This open-source tool was designed to structure the process of comparing SPN performance models, such as the work of (FEITOSA et al., 2025). The source code for Pasid-Validator with autoscaling support is publicly available[2]. The tool has several examples of use and was developed to automate the main validation steps. It can be used to collect times in an experimental environment, simulate the model based on this data, perform empirical execution under different load conditions, and perform statistical analysis of the results. In the collection phase, Pasid-Validator instruments the system to capture intermediate times along the path taken by requests, recording points such as the sending of the request, its receipt, and return. These times reflect the real behavior of the distributed system and are used as a basis for parameterizing the timed transitions of the SPN model. This ensures that the simulation operates under realistic conditions that are compatible with the environment. The tool is also responsible for the continuous generation of requests, simulating a constant load flow, using an exponential distribution.

The component responsible for this sending keeps track of the arrival rate and records, for each request, the sending time and the response time. At the end of the execution, Pasid-Validator calculates the average response time and its standard deviation based on the collected samples, accurately reflecting the observed dynamic behavior. The comparison between the model results and those obtained in the physical experiment is performed automatically by the tool, using statistical adherence tests, with emphasis on the t-test. This process allows for verifying, based on quantitative evidence, whether the simulated values are statistically compatible with the empirical values, within a previously established confidence interval. For this work, because it is open-source, Pasid-Validator was extended with support for autoscaling policies.

The scaling logic has been incorporated directly into the load balancing component: Whenever the occupancy of the request queue exceeds a certain threshold (in this case, 80%), a new service instance is automatically created. This mechanism reflects the policy represented in the SPN model and ensures that the adaptive behavior observed in the simulation is reproduced in the real experiment. With this extension, Pasid-Validator consolidates itself as a tool for validating stochastic models with dynamic characteristics,

[2] <https://github.com/LuisGuilherme-x7/Pasid_Validator_Autoscaling>

offering support for both experimental instrumentation and statistical analysis of the results. Its flexibility and precision allow for validating complex systems under different adaptation strategies, such as reactive scaling and reliability of the results.

7.3.4 Validation Results and Comparative Analysis

The comparison between the SPN model's results and the Pasid-Validator data was performed based on the MRT for different values of arrival rate (AR), defined as the inverse of the arrival delay (AD), i.e., AR = 1/AD. This metric represents the rate of arrival of requests per minute and allows the evaluation of the impact of the load intensity on the system performance. For each AR value analyzed, multiple executions were conducted in the experimental environment, from which the average MRT and the corresponding standard deviation were obtained. The 95% confidence intervals were then calculated based on these samples and used as a reference for the statistical validation of the model results. Table 11 presents the results obtained. It can be observed that, for all load configurations analyzed, the simulated values are within the experimental confidence interval, which indicates statistical adherence between the model and the real system. In addition to statistical consistency, a tendency for MRT to decrease as the arrival rate decreases is observed. This behavior reflects the dynamics expected in systems with adaptive scheduling: With lower load intensity, the system requires fewer instantiations and can operate with fewer active services, maintaining responsiveness within adequate levels.

Tabela 11 – Comparison between experimental and model results for different arrival rates.

Arrival Rate (msg/m)	**Exp. MRT (ms)**	**Std. Dev. (Exp)**	**Model MRT (ms)**	**95% CI (Model)**	**t Value**	**p-Value**
0.017	2049.08	186.28	2105.78	[2071.83; 2139.74]	-0.62	0.540
0.020	2266.25	196.76	2295.20	[2256.23; 2334.17]	-0.48	0.635
0.025	3431.46	311.95	3185.45	[3148.31; 3222.60]	1.33	0.190
0.033	3492.53	317.50	3358.98	[3320.26; 3397.72]	1.12	0.270
0.050	3599.90	327.26	3794.79	[3755.16; 3834.43]	-1.45	0.160

Figure 30 shows the comparison between experimental and modeled Mean Response Time (MRT). At the lowest arrival rate (AR = 0.017 msg/m), the experimental MRT was 2049 ms, while the model predicted 2106 ms, a difference of 2.7%. For AR = 0.020 msg/m, the experimental MRT reached 2266 ms, very close to the model value of 2295 ms. In intermediate loads (AR = 0.025 and 0.033 msg/m), the model predicted 3185 ms and 3359 ms, compared with the experimental 3431 ms and 3493 ms, remaining within 1.3% deviation. At the highest load (AR = 0.050 msg/m), the experimental MRT was 3599 ms, while the model produced 3795 ms, a maximum difference of 5.4%. These results demonstrate that the SPN model consistently follows the experimental curve across all scenarios, maintaining statistical adherence.

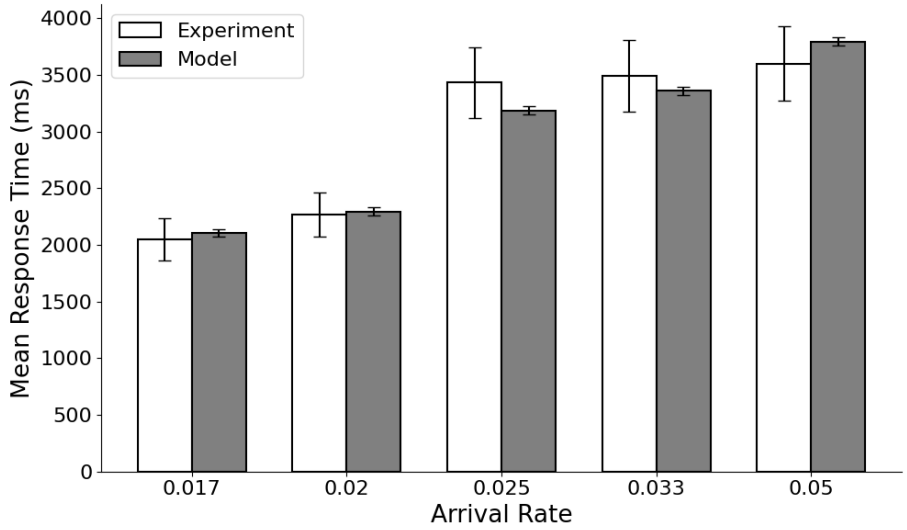


Figura 30 – Comparison between experimental and modeled Mean Response Time (MRT) under different arrival rates. The close alignment across all scenarios confirms the statistical validity and accuracy of the SPN model.

For the lowest load (AR = 0.017 msg/m), the MRT was 2049 ms in the experiment and 2105 ms in the model. As the arrival rate increases, the average response times increase progressively, reaching 3599 ms and 3794 ms, respectively, for AR = 0.050 msg/m. This evolution reflects the increased pressure on the request queue, requiring more actions from the autoscaling mechanism. The most significant difference between the values, approximately 5.4%, occurs under the highest load, and may be the result of startup delays or transient overloads in the instantiated services. Another aspect observed is the reduction in the variability of the experimental measurements with the decrease in AR.

The most significant error ranges appear in the most loaded scenarios, such as AR = 0.050 and 0.033 msg/m, suggesting greater instability under these conditions. In the intermediate scenarios (AR = 0.025 and 0.020 msg/m), the simulated values closely follow the observed ones, with deviations below 1.3%. In AR = 0.017 msg/m, this difference reduces to 2.7%, demonstrating the accuracy of the model under low-demand conditions.

The results obtained confirm that the proposed SPN model faithfully represents the dynamics of the autoscaling system. In vehicular environments, both the tendency for MRT to increase and its variability under different load levels are observed. The proximity between the simulated and experimental values, with differences lower than 5.4% in all scenarios, validates the adopted approach and demonstrates its potential to support performance analysis. In addition, the model's ability to reflect the stabilization of MRT under light loads and the greater fluctuation under high demand indicates that the instancing mechanisms were correctly incorporated.

Figure 31 shows the system behavior graphs under different load conditions, high-

lighting the performance of the implemented dynamic autoscaling mechanism. The top panel displays the evolution of the request arrival rate and the number of active services over time. The system is subjected to load variations, with the AR oscillating between different configured values. In response, the number of service instances is dynamically adjusted, alternating between two, three, and four services as demand increases or decreases. This behavior is a direct result of the implementation of the autoscaling logic in Pasid-Validator, which continuously monitors the occupancy of the request queue and instantiates new service threads whenever utilization exceeds the defined threshold.

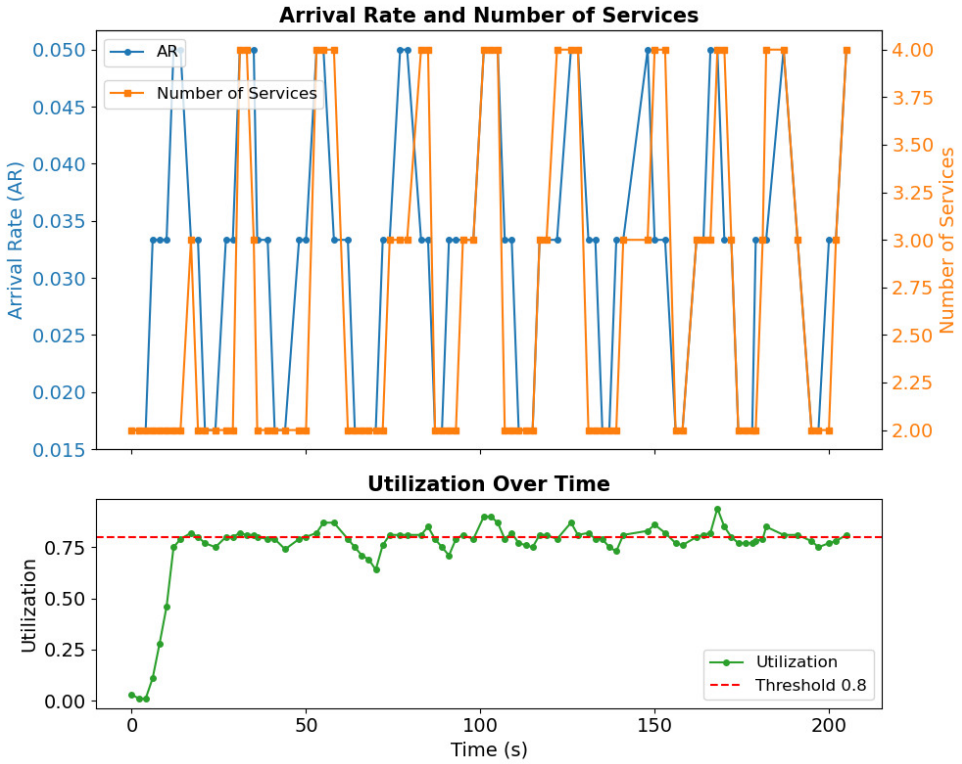


Figura 31 – System behavior during dynamic autoscaling. The top panel shows how service instances adjust with varying load. The bottom panel tracks system utilization, with the 80% threshold acting as a trigger for autoscaling, demonstrating adaptive control stability.

The bottom panel shows the system utilization over time, calculated from the queue and server occupancy. The red dashed line represents the 80% threshold, which acts as a trigger for reactive scheduling. After a brief initial stabilization phase, the system starts to operate mainly around this threshold, with slight natural oscillations. The eviction trigger is implemented based on ongoing underutilization to avoid a precipitous shutdown of active instances. This mechanism allows the system to react prudently to load drops. This behavior indicates that the scheduling control was successful in keeping utilization close to the point of maximum efficiency, without exceeding the operational limits. The coherence between the variations in the arrival rate, the number of allocated services, and

the observed utilization reinforces the fidelity of the implementation in Pasid-Validator, validating the system's adaptive dynamics as represented in the proposed SPN model.

The experimental validation adopted in this work is intentionally aligned with the analytical scope of the models and with the abstraction level required by SPN-based evaluation. The objective is not to reproduce the full operational diversity of vehicular edge infrastructures, but to calibrate service times, scaling latencies, reinstantiation delays, and energy profiles under controlled and traceable conditions. The calibration parameters extracted from these experiments feed the models through measurable and reproducible quantities which is consistent with the methodological framework adopted. Heterogeneous environments, adversarial loads, and fault injections would expand the empirical envelope. Within the defined scope, the chosen experimental setup support model calibration and to demonstrate correspondence between analytical predictions and empirical trends.

7.4 Desing of Experiments

In this section, DoE is applied in conjunction with SPN modeling to quantify the influence of critical variables on the system's energy consumption. The resulting statistical analysis allows for the determination of optimal configuration combinations that reduce energy consumption without compromising performance, offering precise support for efficiency optimization and adaptive resource management in edge computing environments. .

7.4.1 DoE Overview

In the context of this work, DoE is used to evaluate the impact of different factors on the energy efficiency of the VM in system. Table 12 presents the factors and levels used in the simulation. The values chosen for the factors and levels were defined based on an analysis of the operational requirements and limitations of the systems under study. The number of VMs N_C was varied to reflect the difference between low and high computational capacity scenarios, allowing us to observe the impact of scaling on energy consumption. The variation of C_R aims to test different allocations of reserved VMs, which can affect the efficiency and performance of the system. The factors P_R and P_F were adjusted to test different levels of fault tolerance and reinstantiation priorities, with the intention of evaluating how these changes influence the energy consumed. Finally, the size of the queue Q_S, varying, was chosen to simulate different workloads and understand its impact on the energy efficiency of the system.

Table 16 presents the different combinations of factors and levels tested in the simulation, along with the resulting energy consumption values in kilowatt hours for each configuration. By exploring these combinations, DoE allows identifying which configurations

Tabela 12 – Design of experiments.

Factor name	Low Setting	High Setting
N_C	4.0	8.0
C_R	4.0	48.0
P_R	0.1	0.3
P_F	0.1	0.3
Q_S	100.0	1000.0

are most energy efficient and provides valuable insights into system behavior under different operating conditions. Applying DoE not only makes it easier to understand the individual effects of each factor but also helps identify complex interactions between factors, providing a more robust and detailed analysis of the system's energy performance. With this approach, it is possible to optimize system parameters to reduce energy consumption and improve overall efficiency.

Apparent replicates in Table occur because values were rounded to two decimal places due to space constraints in the tabular formatting. In practice, independent replicates of each configuration produced slightly different results, but these differences are not fully visible in the displayed precision. Multiple replicates were calculated for each scenario, and an ANOVA (DAS et al., 2022) test was conducted to confirm the statistical significance of the estimated effects, ensuring that the reported results are robust despite the visual similarities.

7.4.2 Results analysis

The subsection presents the results obtained from the analysis of the effects of the factors and their interactions on the performance of the investigated system. Figure 32 presents the values of the effects, arranged in decreasing order, allowing the identification of the most significant factors and their respective interactions. This analysis sought to understand which elements of the system exert the greatest influence and how they interact to determine the overall behavior. The results show that the C_R factor (with an approximate value of 0.021) is the most significant, indicating that it is the main determinant of the energy performance of the system. This effect shows that C_R should be treated as an important control variable in any optimization or adjustment attempt.

Next, the P_F factor stands out, with an effect close to 0.019, confirming its substantial relevance and suggesting that it plays a central and complementary role to the effect of C_R. The third largest contribution is given by the interaction C_R x Q_S, whose effect is approximately 0.017. This specific interaction implies that the combined dependency contains complex relationships that can impact the system's energy consumption. On the other hand, at the bottom of the graph, we find the interactions P_R x Q_S, N_C x Q_S and P_R x P_F, which present effects of approximately 0.001, 0.002

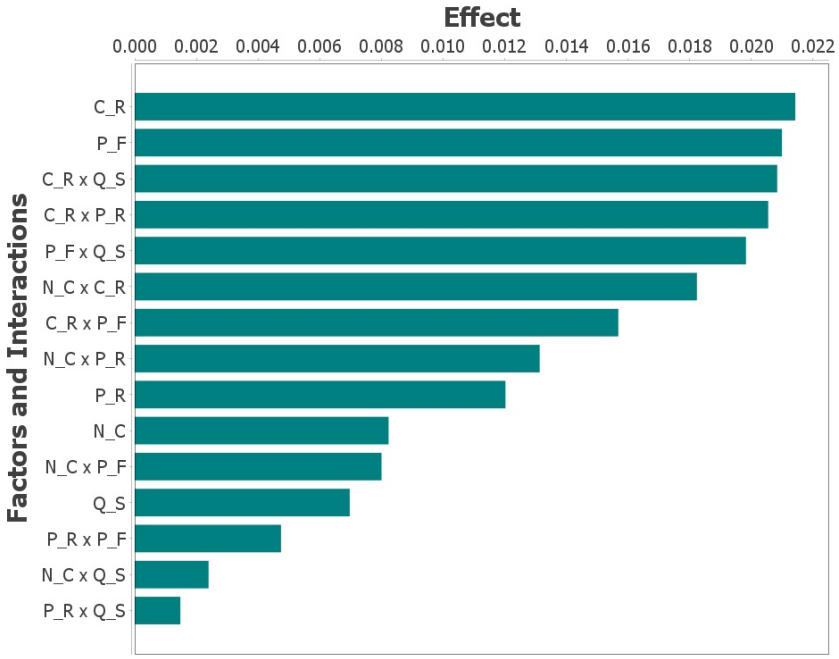
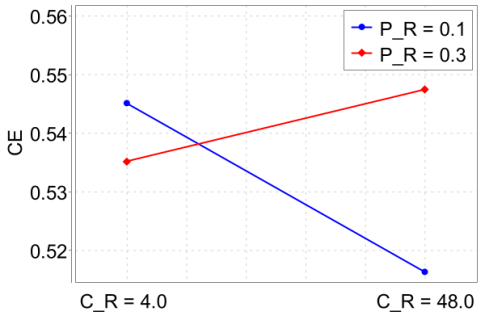


Figura 32 – Bar graph showing the effect of different model factors on energy consumption, based on DoE analysis. Reserved VMs (C_R) and failure weight (P_F) are identified as the most impactful parameters on energy usage.

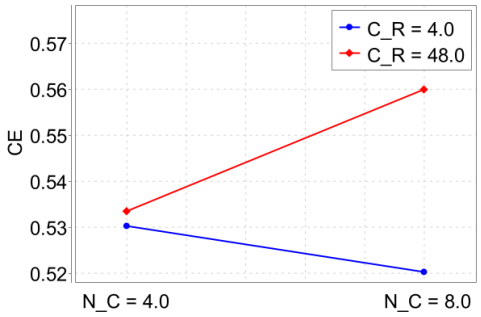
and 0.003, respectively. These values indicate that these interactions have a practically negligible influence on the overall performance of the system. The low value of their effects suggests that they can be discarded in more in-depth analyses or optimization efforts, given that their impact is marginal.

The figure 33 graphically represents the DOE interactions, offering a clear view of how the dynamicity of autoscaling influences the combinations between factors and their impact on the system's Energy Consumption. Figure 33a shows the interaction between the C_R and the reinstantiation weight (P_R) on EC. It can be seen that, for P_R = 0.1 (blue line), increasing C_R from 4 to 48 results in a reduction in energy consumption, suggesting that a greater availability of reserved VMs improves energy efficiency when the reinstantiation weight is low. However, for P_R = 0.3 (red line), the behavior is reversed; a greater number of reserved VMs leads to an increase in energy consumption. This interaction suggests that reinstantiation has a relevant impact on total energy consumption and that its effect depends on the number of available VMs. When the reinstantiation weight is high, there may be a higher cost associated with migrating or restarting VMs, increasing energy consumption as C_R increases. With a low reinstantiation weight, a greater number of reserved VMs can reduce the need for new allocations, optimizing energy use.

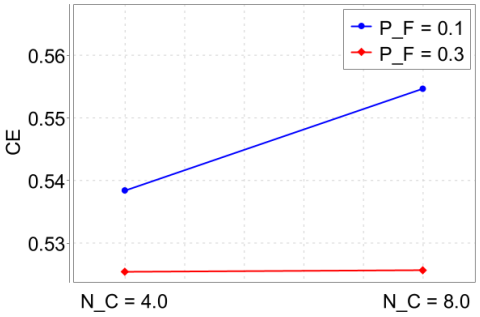
Next, Figure 33b analyzes the interaction between the N_C and the C_R. It can be seen that, for C_R = 4.0 (blue line), the increase in the total number of VMs from 4 to



(a) Reserved VMs x Reinstantiation weight.



(b) Number of VMs x Reserved VMs.



(c) Number of VMs x Fail Weight.

Figura 33 – Interaction graphs between key model parameters and their influence on energy consumption: (a) Reserved VMs (C_R) vs. reinstantiation weight (P_R); (b) Number of VMs (N_C) vs. reserved VMs (C_R); (c) Number of VMs (N_C) vs. failure weight (P_F).

8 results in a slight reduction in energy consumption, possibly due to a better distribution of the processing load, reducing the impact of overload on individual units. In contrast, for C_R = 48.0 (red line), there is a significant increase in energy consumption as the total number of VMs grows. When there are many reserved VMs, the addition of new operational instances can increase energy costs due to the need to maintain multiple active units. The difference in the slope of the lines indicates that the relationship between N_C and CE strongly depends on the number of reserved VMs, reinforcing the importance of an adequate balance between scalability and energy efficiency.

Finally, the Figure 33c presents the relationship between the N_C and the P_F on EC. It can be seen that, for P_F = 0.1 (blue line), increasing N_C from 4 to 8 results in a considerable increase in energy consumption, indicating that. When the probability of failure is low, adding new VMs can generate a greater energy impact due to the cost of maintaining additional instances in continuous operation. In contrast, for P_F = 0.3 (red line), energy consumption remains practically constant, suggesting that a higher failure weight can result in lower VM uptime or higher turnover, reducing total energy consumption. This difference in trends suggests that the impact of autoscaling on energy consumption is important, as it reinforces the need to consider dynamism to improve and

optimize the infrastructure to minimize energy costs.

7.5 Case Studies

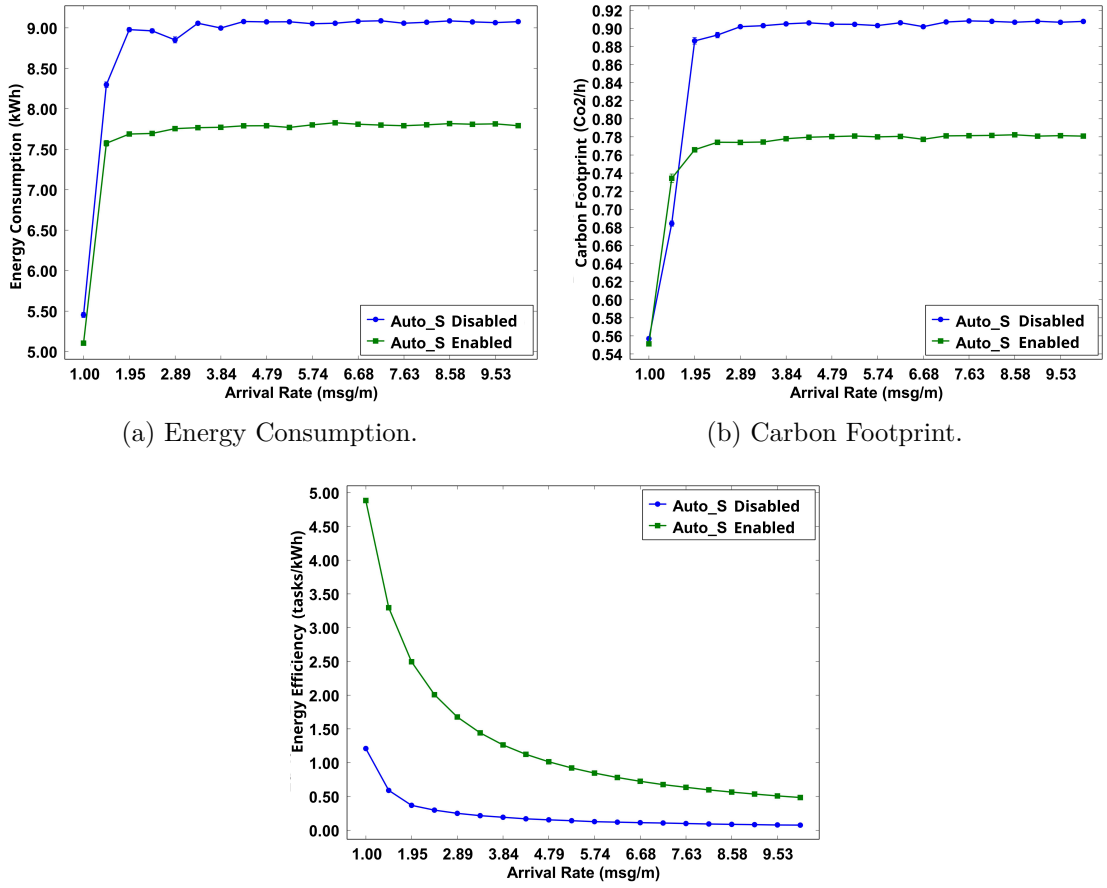
This section presents the results obtained as case studies. For the development of the model and execution of the experiments, the Mercury tool (MACIEL et al., 2017), version 5.0.1, was used. Due to the complexity of the model, the case studies were conducted through stationary simulations, incorporating an approximate error margin of 2%. The values of the system components were based on previously published academic sources, most notably the studies (SILVA et al., 2024; FÉ et al., 2022b). The following two subsections highlight the results obtained through the use of the model. The parameters used for the simulations are presented in Table 13.

Tabela 13 – Parameters used in the model.

Type	Component	Value	Definition
Places	Pqe	0.0	Request waiting queue.
	Ppqe	0.0	Queue of requests to be processed.
	Pcapq	Q_S	Request queue capacity.
	Pprsu	0.0	RSU processing.
	Pcaprsu	N_C	RSU capacity.
	Pcres	C_R	Reserved VMs in the system.
	Pretc	0.0	VM return place.
	Pic	0.0	VM instantiation place.
	Plc	0.0	VM release place.
	Pdf	0.0	Failure detection place.
	Prc	0.0	Reinstantiation place.
Timed Transitions	Tad	AD	Request arrival time.
	Trsu	0.1	RSU arrival transition.
	Ttsrsu	15.7	Service time transition.
	Tic	1.0	VM instantiation transition.
	Trc	2.0	VM reinstantiation transition.
Variables	AD	10.0	Data arrival.
	Q_S	1000.0	Queue size.
	N_C	4.0	Initial number of RSU VMs.
	C_R	25.0	VMs reserved by the system.
	P_F, P_R	0.1, 0.1	Failure and reinstantiation weights.
	E_OP_RSU	0.03	RSU energy consumption.
	E_OP_CONT	0.0001	VM energy consumption.
	Time	60.0	System operation time.
	CEF	0.01	Carbon emission factor.

7.5.1 Case Study 1 - Comparative Analysis of Energy Consumption with and without Autoscaling

This subsection presents the results of the energy impacts of autoscaling on the system's adaptability. One of the main aspects of the work was the prevention of resource waste. Through the autoscaling system, it was possible to reduce energy consumption and minimize the number of idle VMs. In addition, the reinstantiation system stands out, as it contributes to resource savings by correcting failures that occurred during the instantiation of VMs. Figure 34 presents the results obtained, comparing two system configurations: One with autoscaling disabled (Auto_S Disabled) and another with autoscaling enabled (Auto_S Enabled).



(a) Energy Consumption.

(b) Carbon Footprint.

(c) Energy Efficiency.

Figura 34 – Energy impact analysis comparing scenarios with autoscaling enabled vs. disabled: (a) Total energy consumption over varying message arrival rates; (b) Resulting carbon footprint (CO_2/h); (c) Energy efficiency measured in tasks per kWh. Autoscaling consistently shows better performance and sustainability.

Figure 34a shows the relationship between arrival rate (msg/m) and energy consumption (kWh). At 1.00 msg/m, Auto_S Enabled is approximately 5.1 kWh and Auto_S Disabled approximately 8.3 kWh, a reduction of about 39% with autoscaling. At 1.95

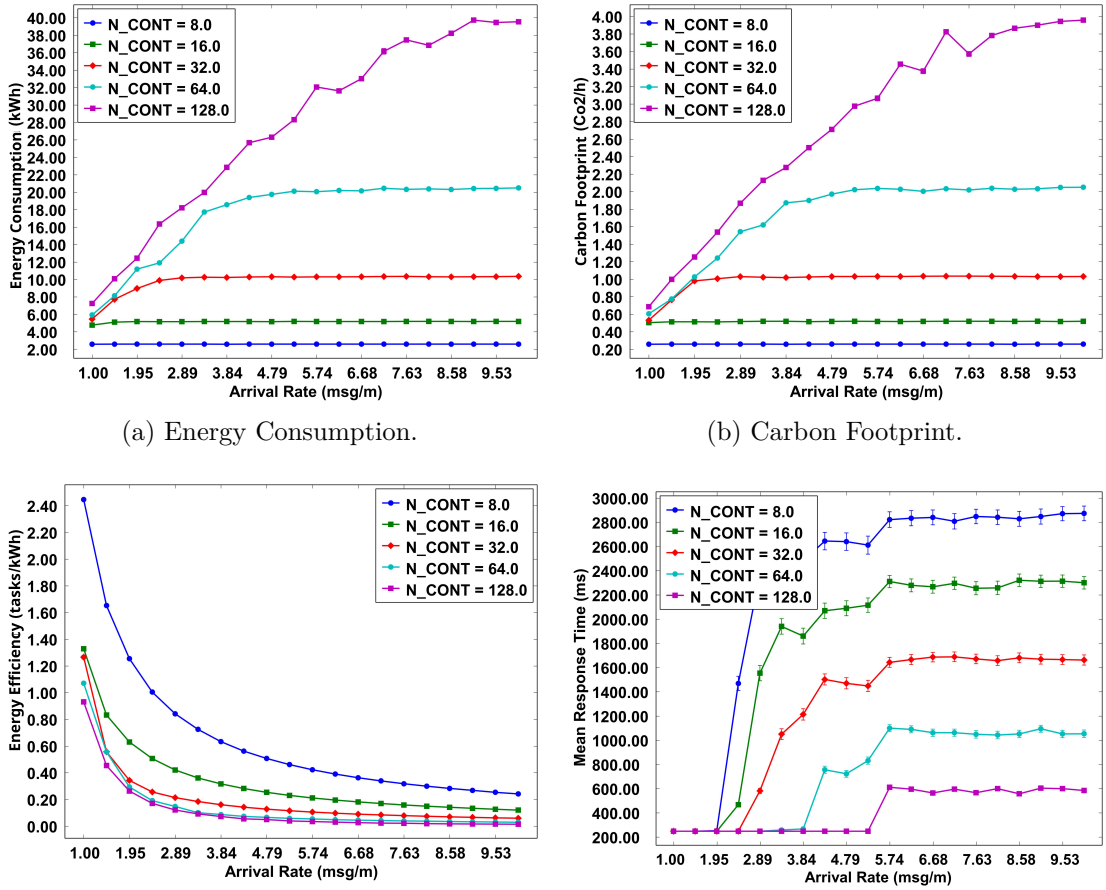
msg/m, consumption rises to approximately 7.7 kWh (enabled) compared to approximately 8.9 kWh (disabled), about 14% lower with autoscaling. At 2.89 msg/m, values are approximately 7.8 kWh (enabled) in contrast to approximately 8.9 kWh (disabled), a reduction close to 12%. For medium-to-high loads (3.84–9.53 msg/m), the non-autoscaling curve stays near 9.0–9.1 kWh, while autoscaling remains around 7.8–8.0 kWh. For example, at 7.63 msg/m it is approximately 7.98 kWh (enabled) compared with approximately 9.06 kWh (disabled), about 12% lower; at 9.53 msg/m, approximately 7.9 kWh relative to approximately 9.05 kWh, about 13% lower. Overall, autoscaling yields consistent savings of 12–14% under medium loads, with a larger gain (39%) at very low load due to deactivation of idle capacity.

Figure 34b shows the behavior of the system's Carbon Footprint (CO2/h) as a function of the arrival rate. At 1.00 msg/m, the curves are essentially equal, with Auto_S Enabled approximately 0.56 CO2/h and Auto_S Disabled approximately 0.54 CO2/h (a residual difference at very low load). From 1.95 msg/m onward, the advantage of autoscaling becomes evident: approximately 0.75 CO2/h (enabled) compared to approximately 0.89 CO2/h (disabled), a reduction of about 16%. At 2.89 msg/m the values are approximately 0.76 (enabled) in contrast to approximately 0.90 (disabled), again about 16% lower with autoscaling. For medium to high loads (3.84–9.53 msg/m), the non-autoscaling curve remains around 0.90–0.92 CO2/h, while autoscaling stays near 0.78–0.81 CO2/h. For example, at 7.63 msg/m it is approximately 0.80 (enabled) compared with approximately 0.91 (disabled), about 12% lower; and at 9.53 msg/m, approximately 0.80 relative to approximately 0.91, about 13% lower. In summary, except at the lowest load point (1.00 msg/m), autoscaling consistently reduces the carbon footprint by roughly 12–16% across the evaluated range.

Figure 34c illustrates the behavior of Energy Efficiency (tasks/kWh) as a function of the arrival rate. At 1.00 msg/m, Auto_S Enabled is approximately 4.9 tasks/kWh, while Auto_S Disabled is approximately 1.3 tasks/kWh (about 3.8× higher with autoscaling). As the rate increases to 1.95 msg/m, values are approximately 3.4 (enabled) compared to approximately 0.5 (disabled), about 6.8× higher. At 2.89 msg/m, approximately 2.5 relative to approximately 0.25 (about 10×). At 3.84 msg/m, approximately 1.9 compared with approximately 0.17 (about 11×). At 4.79 msg/m, approximately 1.6 in contrast to approximately 0.12 (about 13×). At 5.74 msg/m, approximately 1.3 compared to approximately 0.10 (about 13×). At 6.68 msg/m, approximately 1.1 relative to approximately 0.08 (about 14×). At 7.63 msg/m, approximately 0.95 compared with approximately 0.07 (about 14×). At 8.58 msg/m, approximately 0.85 in contrast to approximately 0.06 (about 14×). At 9.53 msg/m, approximately 0.78 compared with approximately 0.05 (about 15×). Both curves decrease as load grows, but autoscaling preserves substantially higher efficiency across the entire range.

7.5.2 Case Study 2 - Influence of the Number of VMs on Energy Performance

In this subsection, the variation in the number of VMs in the system is used to analyze the energy performance results. Throughout the analysis, it was observed that the number of VMs appears as one of the main factors. Therefore, the autoscaling system was used to vary the number of VMs available. Figure 35 presents the results of this variation in the number of VMs in search of the best configurations, thus promoting greater sustainability and energy performance of the system.



(a) Energy Consumption.

(b) Carbon Footprint.

(c) Energy Efficiency.

(d) Mean Response Time.

Figura 35 – Evaluation of how VM quantity affects system performance and sustainability: (a) Energy consumption across message arrival rates; (b) Carbon footprint for each configuration; (c) Energy efficiency; (d) Mean Response Time (MRT). The results illustrate trade-offs between scalability and energy-aware operation.

Figure 35a presents the relationship between energy consumption (in kWh) and message arrival rate (msg/m) for different values of N_CONT, representing the number of VMs in the system. The results show that energy consumption increases significantly with the number of VMs, especially at higher arrival rates. For N_CONT=8, consumption is the lowest among the scenarios, remaining constant and below 0.4 kWh, even with high arrival rates. The scenario with N_CONT=16 presents slightly higher consumption, but stable at around 0.6kWh, while for N_CONT=32, consumption increases slightly, stabilizing

at approximately 0.8kWh. In the cases of N_CONT=64 and N_CONT=128, a different behavior is observed. With N_CONT=64, the initial consumption is around 1.0 kWh, gradually increasing until stabilizing around 1.8 kWh. The scenario with N_CONT=128 records the highest consumption, starting at 1.5 kWh and increasing almost linearly up to 3.5 kWh at the highest rates. The results indicate that the increase in the number of VMs is directly associated with higher energy consumption. In scenarios with low demand (reduced msg/m rates), using fewer VMs may be more energy efficient. However, in situations of high demand, increased consumption is inevitable due to the need for greater capacity to handle the request load.

Figure 35b examines the carbon footprint (in CO2/h) in relation to the message arrival rate (msg/m) for different amounts of VMs. For N_CONT =8, the carbon footprint remains stable around 0.2 CO2/h, indicating environmental efficiency with fewer VMs. For N_CONT=16 and N_CONT=32, the environmental impact increases slightly as the arrival rate increases, ranging from 0.4 to 0.8 CO2/h. With N_CONT=64, the ecological implications grow more significantly, exceeding 2 CO2/h when the arrival rate exceeds 5 msg/m. In the case of N_CONT=128, a sharp increase is observed, reaching 4 CO2/h at the highest rates, evidencing an exponential growth in emissions. Although configurations with many VMs, such as N_CONT=128, are suitable for high arrival rates, this comes at the cost of a considerably higher environmental impact. Configurations with fewer VMs, such as N_CONT=8, are more sustainable, but may not meet demand in high-load scenarios. Therefore, it is necessary to balance performance and environmental sustainability.

Figure 35c presents the energy efficiency results in tasks/kWh for different VM configurations. N_CONT=8, the efficiency is higher at low arrival rates, starting at 2.4 tasks/kWh and decreasing to less than 0.4 tasks/kWh at rates above 5 msg/m. For N_CONT=16, the initial efficiency is lower, starting at 1.2 tasks/kWh and stabilizing around 0.4 tasks/kWh at high rates. In the cases of N_CONT=32, 64, and 128, the initial efficiency values are even lower, dropping quickly to less than 0.2 tasks/kWh at intermediate rates. The results indicate that energy efficiency decreases as the number of VMs increases, especially in high-load scenarios. Although configurations with few VMs, such as N_CONT=8, are more energy efficient, they may not meet high demands. On the other hand, configurations with many VMs, such as N_CONT=128, are low energy efficient even under high-load conditions, reflecting a less sustainable use of energy.

Figure 35d illustrates the variation of the Mean Response Time (MRT) as the message arrival rate (msg/m) increases, considering different numbers of VMs allocated in the system. The observed behavior highlights the direct impact of scalability on the infrastructure's responsiveness. In scenarios with a lower number of VMs (N_CONT = 8.0 and 16.0), the MRT grows rapidly after the arrival rate exceeds 2.0 msg/m. For N_CONT = 8.0, for example, the mean response time exceeds 2600 ms from 2.89 msg/m,

demonstrating an apparent saturation of the system. The same pattern is repeated for N_CONT = 16.0, although with a more damped curve. On the other hand, more robust configurations (N_CONT = 64.0 and 128.0) maintain the MRT at low and stable levels even under high loads. In the case of N_CONT = 128.0, the system maintains the MRT below 500 ms throughout the entire arrival range analyzed, demonstrating high resilience to overload. This demonstrates the effectiveness of the autoscaling mechanism, as it shows that the greater the availability of VMs, the greater the system's ability to maintain responsiveness in the face of abrupt variations in demand.

8 Conclusion

This work presented a formal approach to model, analyze, and validate the availability, performance, and sustainability of VANET architectures integrated with edge computing, based on Stochastic Petri Nets. The proposed model allows for the evaluation of system behavior under faults, load variations, and adaptation policies, without the need for extensive validation in real-world environments. The combination of virtual machine migration, autoscaling, and energy metrics enables the joint analysis of service continuity, response time, and energy consumption. With this, the results obtained allow for the comparison of configuration alternatives and support for operational decisions in distributed vehicular infrastructures.

The availability analysis incorporated physical, logical, and communication failures, as well as machine migration mechanisms, which are crucial for the operational continuity of virtual RSUs. The model demonstrated that, in scenarios with simultaneous failures, migration reduces the average recovery time, keeping the system functional during degradation events. The migration mechanism improved availability by dynamically redistributing instances during failure events, ensuring service continuity. The availability model confirmed that VM migration mechanisms significantly increase the availability of RSUs, raising it from approximately 10–14 *nines* to about 20 *nines*, by reducing the impact of physical, logical, and network failures. This gain results from the dynamic reallocation of instances during failure events, which reduces downtime and accelerates service recovery. Consequently, the model allows for a quantitative assessment of when migration becomes necessary to maintain high levels of operational continuity.

Performance evaluation demonstrated that horizontal autoscaling and conditional reinstantiation policies improve stability under variable load. The stationary model revealed that average VM utilization tends toward equilibrium close to 80%, with a 30% reduction in average response time compared to the scenario without elasticity. The performance model also accurately captured the dynamics of saturation, queuing, and VM reinstantiation, showing that balanced configurations maintain an average response time close to 700 ms, while undersized systems exceed 1100 ms. These results highlight clear operational boundaries between stability and degradation, allowing the identification of load and capacity combinations that avoid persistent congestion. Therefore, the model assists in defining scaling policies compatible with latency requirements and under variable demand.

The sustainability analysis integrated energy metrics and confirmed that adaptive behavior also implies environmental gains. The inclusion of automatic scaling policies reduced average energy consumption by approximately maintaining savings between 12%

and 14%, without a perceptible impact on response time. The technical contributions of this work focus on three axes: integrated formal modeling, empirical calibration, and experimental validation. This behavior results from the controlled reduction in the number of active VMs, aligning energy consumption with effective demand. Thus, the model allows for the evaluation of the energy impact of different operational policies over time. The adjustment methodology, supported by empirical and experimental data, ensured quantitative consistency between simulation and observation.

The validation confirmed the accuracy of the SPN model in capturing system behavior and evaluating the trade-offs between availability and energy efficiency. The SPN performance model was validated with experimental data, showing high adherence between the modeled MRTs and the values observed by the Pasid-Validator, with all p-values above 0.05. This result confirms that the model consistently reproduces the behavior observed in the real environment, including under different load regimes. The statistical validation reinforces the reliability of the model as a predictive tool for performance analysis. As a future perspective, it is recommended to extend the model to hierarchical architectures involving cloud, edge, and vehicles, incorporating latencies and synchronization, and machine learning models. It is also suggested to use hybrid SPN-Markov models to capture temporal dependencies and correlated events, as well as multi-objective optimization techniques to balance availability, latency, and energy. In summary, this dissertation demonstrated that Stochastic Petri Nets are suitable for representing, analyzing, and predicting the behavior of VANETs.

Referências

AGARWAL, S.; DAS, A.; DAS, N. An efficient approach for load balancing in vehicular ad-hoc networks. In: *2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. [S.l.: s.n.], 2016. p. 1–6. 26, 27

AHMED, I. Z.; MOHAMED, T. M.; SADEK, R. A. Challenges of vehicular ad hoc networks: Egypt perspective. In: *Second International Conference for Computing and Informatics, ICCI*. [S.l.: s.n.], 2017. 12, 13

ALI, G. M. N.; CHAN, E.; LI, W. On scheduling data access with cooperative load balancing in vehicular ad hoc networks (vanets). *The Journal of Supercomputing*, Springer, v. 67, p. 438–468, 2014. 26, 27, 28

ALI, G. M. N. et al. An efficient cooperative load balancing approach in rsu-based vehicular ad hoc networks (vanets). In: IEEE. *2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2014)*. [S.l.], 2014. p. 52–57. 26, 27

ALVARENGA, L. D. C.; SOUSA, P.; COSTA, A. Allocation and migration of microservices in sdn-based vehicular fog networks. In: *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*. [S.l.: s.n.], 2022. p. 1–4. 21, 24

ANDRADE, E.; NOGUEIRA, B. Dependability evaluation of a disaster recovery solution for iot infrastructures. *The Journal of Supercomputing*, Springer, v. 76, n. 3, p. 1828–1849, 2020. 44, 60

ANWER, M. S.; GUY, C. A survey of vanet technologies. *Journal of Emerging Trends in Computing and Information Sciences*, Citeseer, v. 5, n. 9, p. 661–671, 2014. 4

AOKI, J. et al. Experimental evaluation of location-based mobile agent migration mechanism for information dissemination scheme in vanets. In: *2015 Third International Symposium on Computing and Networking (CANDAR)*. [S.l.: s.n.], 2015. p. 271–274. 20, 24

APM, A. P. d. M. *Brasil é o Terceiro País com Mais Mortes de Trânsito - APM*. 2022. <https://www.apm.org.br/ultimas-noticias/brasil-e-o-terceiro-pais-com-mais-mortes-de-transito/>. Acesso em: 8 jan. 2024. 3

ARAÚJO, G. et al. Vehicular cloud computing networks: Availability modelling and sensitivity analysis. *International Journal of Sensor Networks*, Inderscience Publishers (IEL), v. 36, n. 3, p. 125–138, 2021. 4, 5

ARAÚJO, G. et al. Vehicular cloud computing networks: Availability modelling and sensitivity analysis. *International Journal of Sensor Networks*, Inderscience Publishers (IEL), v. 36, n. 3, p. 125–138, 2021. 44

AUDU, G. A. et al. Reliability and quality of service of an off-grid wind powered roadside unit in a motorway vehicular environment. *Vehicular Communications*, Elsevier, v. 9, p. 176–187, 2017. 44

BALZANO, W.; STRANIERI, S. Data dissemination in vehicular ad hoc network: a model to improve network congestion. In: *Web, Artificial Intelligence and Network Applications: Proceedings of the Workshops of the 33rd International Conference on Advanced Information Networking and Applications (WAINA-2019)*. [S.l.]: Springer, 2019. p. 851–859. 11, 12

Blessy A, M. C.; S, B. Maximizing vanet performance in cluster head selection using intelligent fuzzy bald eagle optimization. *Vehicular Communications*, p. 100660, 2023. ISSN 2214-2096. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2214209623000906>. 4

BOUKHALFA, F. et al. Performance evaluation of an active signaling based time-slot scheduling scheme for connected vehicles. *Annals of Telecommunications*, Springer, v. 76, p. 363–374, 2021. 11

BRITO, C. et al. Avaliação de disponibilidade de uma arquitetura de computação de borda com redes de petri estocásticas. In: *Anais do XIX Workshop em Desempenho de Sistemas Computacionais e de Comunicação*. Porto Alegre, RS, Brasil: SBC, 2020. p. 175–180. ISSN 2595-6167. Disponível em: <https://sol.sbc.org.br/index.php/wperformance/article/view/11117>. 24

BRITO, C. et al. Offloading data through unmanned aerial vehicles: a dependability evaluation. *Electronics*, MDPI, v. 10, n. 16, p. 1916, 2021. 14

CAMPOLONGO, F.; TARANTOLA, S.; SALTELLI, A. Tackling quantitatively large dimensionality problems. *Computer Physics Communication*, Institute Jbr Systems, lnformatics and Safety. Joint Research Centre of the European Commission, v. 117, n. 1, p. 75–85, 1999. 16

CARDOTE, A.; SARGENTO, S.; STEENKISTE, P. On the connection availability between relay nodes in a vanet. In: *2010 IEEE Globecom Workshops*. [S.l.: s.n.], 2010. p. 181–185. 21, 24

CHEN, L.; HA, W. Reliability prediction and qos selection for web service composition. *International Journal of Computational Science and Engineering*, Inderscience Publishers (IEL), v. 16, n. 2, p. 202–211, 2018. 14

CHOUDHARY, S.; DORLE, S. Empirical investigation of vanet-based security models from a statistical perspective. In: *2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)*. [S.l.: s.n.], 2021. p. 1–8. 3

COHERENCE. *10 Best Practices for Effective Auto-Scaling on AWS*. 2023. Accessed: 2025-05-05. Disponível em: <https://www.withcoherence.com/articles/10-best-practices-for-effective-auto-scaling-on-aws>. 78

COSTA, I. et al. Availability evaluation and sensitivity analysis of a mobile backend-as-a-service platform. *Quality and Reliability Engineering International*, Wiley Online Library, v. 32, n. 7, p. 2191–2205, 2016. 16

CUMBAL, R. et al. Optimal resources allocation from vanet infrastructures in dynamic mobile environments. In: IEEE. *2019 IEEE Latin-American Conference on Communications (LATINCOM)*. [S.l.], 2019. p. 1–5. 20, 23

CUNHA, B. et al. Smart traffic control in vehicle ad-hoc networks: a systematic literature review. *International Journal of Wireless Information Networks*, Springer, v. 28, n. 3, p. 362–384, 2021. 12, 13

DAS, B. K. et al. Analysis of variance (anova) and design of experiments. In: *Concept Building in Fisheries Data Analysis*. [S.l.]: Springer, 2022. p. 119–136. 84

FÉ, I. et al. Performance-cost trade-off in auto-scaling mechanisms for cloud computing. *Sensors*, MDPI, v. 22, n. 3, p. 1221, 2022. 5

FÉ, I. et al. Performance-cost trade-off in auto-scaling mechanisms for cloud computing. *Sensors*, MDPI, v. 22, n. 3, p. 1221, 2022. 87

FEITOSA, L. et al. A comprehensive performance evaluation of container migration strategies. *Computing*, Springer, v. 107, n. 2, p. 1–39, 2025. 79

FEITOSA, L. et al. Performance evaluation of message routing strategies in the internet of robotic things using the d/m/c/k/fcfs queuing network. *Electronics*, MDPI, v. 10, n. 21, p. 2626, 2021. 4, 16

Ferreira, W. *IBM alerta sobre pressão de dados dos carros conectados*. 2023. Disponível em: https://www.telesintese.com.br/ibm-alerta-sobre-pressao-dos-carros-conectados-nas-redes-de-telecom/. Acesso em: 09 de agosto 2023. 3

GAO, H. et al. V2vr: reliable hybrid-network-oriented v2v data transmission and routing considering rsus and connectivity probability. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 22, n. 6, p. 3533–3546, 2020. 12, 13, 14

GERMAN, R. *Performance analysis of communication systems - modelling with non-Markovian stochastic Petri nets*. [S.l.]: Wiley, 2000. (Wiley-Interscience series in systems and optimization). ISBN 978-0-471-49258-0. 14, 15

GOYAL, A. K.; TRIPATHI, A. K.; AGARWAL, G. Security attacks, requirements and authentication schemes in vanet. In: *2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*. [S.l.: s.n.], 2019. v. 1, p. 1–5. 20, 24

HAIDER, S. E.; KHAN, M. F.; SAEED, Y. Adaptive load balancing approach to mitigate network congestion in vanets. *Computers*, v. 13, n. 8, 2024. ISSN 2073-431X. Disponível em: <https://www.mdpi.com/2073-431X/13/8/194>. 26, 27

HASHIMOTO, T. et al. A mobile agent migration mechanism for information dissemination scheme in vanets considering entrance and exit of mobile nodes. In: *2015 IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*. [S.l.: s.n.], 2015. p. 89–94. 24

HASROUNY, H. et al. Vanet security challenges and solutions: A survey. *Vehicular Communications*, Elsevier, v. 7, p. 7–20, 2017. 3

HOTA, L. et al. A performance analysis of vanets propagation models and routing protocols. *Sustainability*, MDPI, v. 14, n. 3, p. 1379, 2022. 11, 12

HOZOURI, A. et al. An overview of vanet vehicular networks. *arXiv preprint arXiv:2309.06555*, 2023. 11

HUSSAIN, R.; HUSSAIN, F.; ZEADALLY, S. Integration of vanet and 5g security: A review of design and implementation issues. *Future Generation Computer Systems*, Elsevier, v. 101, p. 843–864, 2019. 4

ISMAIL, N. et al. Enhanced congestion control model based on message prioritization and scheduling mechanism in vehicle-to-infrastructure (v2i). In: *Journal of Physics: Conference Series*. [S.l.]: IOP Publishing, 2022. v. 2312, n. 1, p. 012087. 12, 13

JAIN, R. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. [S.l.]: John Wiley & Sons, 1990. 54

JAIN, R. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. [S.l.]: John Wiley & Sons, 1990. 74

JEONG, J. et al. Toms: Tcp context migration scheme for efficient data services in vehicular networks. In: *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*. [S.l.: s.n.], 2017. p. 360–364. 20, 24

K, S.; CHINNASAMY, C. Efficient vanet handover scheme using ssdn by incorporating media independent handover framework. *Measurement: Sensors*, v. 26, p. 100684, 2023. ISSN 2665-9174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S266591742300020X>. 21, 24

KARABULUT, M. A. et al. Inspecting vanet with various critical aspects–a systematic review. *Ad Hoc Networks*, Elsevier, p. 103281, 2023. 5

KASSIR, S. et al. An analytical model and performance evaluation of multihomed multilane vanets. *IEEE/ACM Transactions on Networking*, IEEE, v. 29, n. 1, p. 346–359, 2020. 11, 13

KHAN, S. et al. Security challenges of location privacy in vanets and state-of-the-art solutions: A survey. *Future Internet*, MDPI, v. 13, n. 4, p. 96, 2021. 12, 13

KHAN, Z.; ABBAS, F.; HAMIDULLAH. A conceptual framework of virtualization and live-migration for vehicle to infrastructure (v2i) communications. In: *2019 IEEE 11th International Conference on Communication Software and Networks (ICCSN)*. [S.l.: s.n.], 2019. p. 590–594. 20, 24

LENDREM, D.; OWEN, M.; GODBERT, S. Doe (design of experiments) in development chemistry: Potential obstacles. *Organic Process Research & Development*, ACS Publications, v. 5, n. 3, p. 324–327, 2001. 57

LOBO, A. et al. Expolynomial modelling for supporting vanet infrastructure planning. In: *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*. [S.l.: s.n.], 2017. p. 86–91. 21, 24

MACIEL, P. et al. Mercury: Performance and dependability evaluation of systems with exponential, expolynomial, and general distributions. In: IEEE. *2017 IEEE 22nd Pacific Rim international symposium on dependable computing (PRDC)*. [S.l.], 2017. p. 50–57. 87

MACIEL, P. R.; LINS, R. D.; CUNHA, P. R. *Introdução às redes de Petri e aplicações*. [S.l.]: UNICAMP-Instituto de Computacao Sao Paulo, Brazil, 1996. 55

MACIEL, P. R. M. *Performance, reliability, and availability evaluation of computational systems, Volume 2: Reliability, availability modeling, measuring, and data analysis*. [S.l.]: CRC Press, 2023. 14, 31

MACIEL, P. R. M. *Performance, reliability, and availability evaluation of computational systems, volume I: performance and background*. [S.l.]: CRC Press, 2023. 14

MAHMOOD, D. A.; HORVáTH, G. Analysis of the message propagation speed in vanet with disconnected rsus. *Mathematics*, MDPI, v. 8, n. 5, p. 782, 2020. 11, 13

MALIK, A.; PANDEY, B. Security analysis of discrete event based threat driven authentication approach in vanet using petri nets. *Int. J. Netw. Secur.*, v. 20, n. 4, p. 601–608, 2018. 20, 24

MANIVANNAN, D.; MONI, S. S.; ZEADALLY, S. Secure authentication and privacy-preserving techniques in vehicular ad-hoc networks (vanets). *Vehicular Communications*, Elsevier, v. 25, p. 100247, 2020. 3

MARSAN, M. A. et al. *Modelling with Generalized Stochastic Petri Nets*. USA: John Wiley & Sons, Inc., 1994. ISBN 0471930598. 14

MARSAN, M. A. et al. Modelling with generalized stochastic petri nets. *ACM SIGMETRICS performance evaluation review*, ACM New York, NY, USA, v. 26, n. 2, p. 2, 1998. 15

MARTIN-FAUS, I. V. et al. Transient analysis of idle time in vanets using markov-reward models. *IEEE Transactions on Vehicular Technology*, v. 67, n. 4, p. 2833–2847, 2018. 20, 24

MAZHAR, S. et al. *State-of-the-art authentication and verification schemes in VANETs: a survey. Veh. Commun. 49, 100804 (2024)*. 2024. 11

MCHERGUI, A.; MOULAHI, T.; NASRI, S. Baas: Broadcast as a service cross-layer learning-based approach in cloud assisted vanets. *Computer Networks*, v. 182, p. 107468, 2020. ISSN 1389-1286. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128620311464>. 21, 24

MEHTA, T.; MAHATO, D. P. Effective scheduling and nature inspired hybrid load balancing in vanets. In: *8th International Conference on Computing in Engineering and Technology (ICCET 2023)*. [S.l.: s.n.], 2023. v. 2023, p. 266–273. 26, 27

MELO, C. et al. Capacity-oriented availability model for resources estimation on private cloud infrastructure. In: IEEE. *2017 IEEE 22nd Pacific rim international symposium on dependable computing (PRDC)*. [S.l.], 2017. p. 255–260. 44, 60

MOHAMMED, N. A. et al. Resource allocation with energy balancing for uavs assisted vanets based intelligent transportation system. In: *2023 Al-Sadiq International Conference on Communication and Information Technology (AICCIT)*. [S.l.: s.n.], 2023. p. 282–287. 26

MOLLOY, M. K. Performance analysis using stochastic petri nets. *IEEE Trans. Comput.*, IEEE Computer Society, Washington, DC, USA, v. 31, p. 913–917, September 1982. ISSN 0018-9340. 14

MOUSA, R. J.; HUSZáK, ; SALMAN, M. A. Enhancing vanet connectivity through a realistic model for rsu deployment on highway. In: *Journal of Physics: Conference Series*. [S.l.]: IOP Publishing, 2021. v. 1804, n. 1, p. 012104. 12

MURATA, T. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, IEEE, v. 77, n. 4, p. 541–580, 1989. 14

NARESH, R. et al. A routing in vanet towards smart business cities using optimization techniques. In: *Digital Twin Technology and AI Implementations in Future-Focused Businesses*. [S.l.]: IGI Global, 2024. p. 1–13. 5

NI, Y.; ZHAO, C.; CAI, L. Hybrid rsu management in cybertwin-iov for temporal and spatial service coverage. *IEEE Transactions on Vehicular Technology*, IEEE, v. 71, n. 5, p. 4596–4606, 2021. 3

OUHMIDOU, H. et al. Traffic control, congestion management and smart parking through vanet, ml, and iot: A review. In: IEEE. *2023 10th International Conference on Wireless Networks and Mobile Communications (WINCOM)*. [S.l.], 2023. p. 1–6. 5

PARASHAR, S.; TIWARI, R. Traffic control and qos improvement analysis in v-to-v and v-to-rsu communication in vanet. In: IEEE. *2023 World Conference on Communication & Computing (WCONF)*. [S.l.], 2023. p. 1–5. 5

QIONG, W. et al. Towards v2i age-aware fairness access: a dqn based intelligent vehicular node training and test method. *Chinese Journal of Electronics*, CIE, v. 32, n. 6, p. 1230–1244, 2023. 5

QUN, R.; AREFZADEH, S. M. A new energy-aware method for load balance managing in the fog-based vehicular ad hoc networks (vanet) using a hybrid optimization algorithm. *IET Communications*, v. 15, n. 13, p. 1665–1676, 2021. Disponível em: <https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/cmu2.12179>. 26, 27

RAJENDRAN, N. et al. Delay-tolerant load balancing routing model for rsu-assisted distributed vehicular adhoc networks. In: *2023 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*. [S.l.: s.n.], 2023. p. 1–7. 25, 26

RAUT, C. M.; DEVANE, S. R. Intelligent transportation system for smartcity using vanet. In: IEEE. *2017 International Conference on Communication and Signal Processing (ICCSP)*. [S.l.], 2017. p. 1602–1605. 4

RAZA, A. et al. An uav-assisted vanet architecture for intelligent transportation system in smart cities. *International Journal of Distributed Sensor Networks*, v. 17, n. 7, p. 15501477211031750, 2021. 25, 26

RODRIGUES, L. et al. Modelo estocástico para avaliaçao de disponibilidade de hospitais inteligentes. In: SBC. *Anais do XIX Workshop em Desempenho de Sistemas Computacionais e de Comunicação*. [S.l.], 2020. p. 145–156. 14

SANTOS, L. et al. Data processing on edge and cloud: a performability evaluation and sensitivity analysis. *Journal of Network and Systems Management*, Springer, v. 29, n. 3, p. 27, 2021. 16

SETHI, V.; PAL, S.; VYAS, A. Online energy-efficient scheduling algorithm for renewable energy-powered roadside units in vanets. In: *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. [S.l.: s.n.], 2020. p. 506–514. 26, 27

SHAH, A. S.; ILHAN, H.; TURELI, U. Modeling and performance analysis of the ieee 802.11 p mac for vanets. In: IEEE. *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*. [S.l.], 2019. p. 393–396. 20, 24

SHEN, J. et al. Secure real-time traffic data aggregation with batch verification for vehicular cloud in vanets. *IEEE Transactions on Vehicular Technology*, IEEE, v. 69, n. 1, p. 807–817, 2019. 4

SIDDIQI, M. H. et al. Dynamic priority-based efficient resource allocation and computing framework for vehicular multimedia cloud computing. *IEEE access*, IEEE, v. 8, p. 81080–81089, 2020. 20, 24

SILVA, L. et al. Desvendando a elasticidade de máquinas virtuais em vanets: Uma estratégia para aperfeiçoar o planejamento de capacidade em rsus. In: *Anais do XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. Porto Alegre, RS, Brasil: SBC, 2024. p. 169–182. ISSN 2177-9384. Disponível em: <https://sol.sbc.org.br/index.php/sbrc/article/view/29791>. 87

SILVA, L. G. et al. Dynamically adaptive rsus in vanets: An approach focused on sustainability. *International Journal of Communication Systems*, Wiley Online Library, v. 38, n. 16, p. e70283, 2025. 6

SILVA, L. G. et al. Desvendando a elasticidade de máquinas virtuais em vanets: Uma estratégia para aperfeiçoar o planejamento de capacidade em rsus. In: SBC. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*. [S.l.], 2024. p. 169–182. 6, 12

SILVA, L. G. et al. Urban advanced mobility dependability: A model-based quantification on vehicular ad hoc networks with virtual machine migration. *Sensors*, MDPI, v. 23, n. 23, p. 9485, 2023. 6, 20

SINGH, G. D. et al. Hybrid genetic firefly algorithm-based routing protocol for vanets. *IEEE Access*, IEEE, v. 10, p. 9142–9151, 2022. 3

TAHER, Y. H. et al. Performance evaluation of software defined networking into vanets system. *Bulletin of Electrical Engineering and Informatics*, v. 13, n. 5, p. 3770–3778, 2024. 11, 12

TANG, Y. et al. Delay-minimization routing for heterogeneous vanets with machine learning based mobility prediction. *IEEE Transactions on Vehicular Technology*, IEEE, v. 68, n. 4, p. 3967–3979, 2019. 20, 23

VERMA, R. An efficient secure vanet communication using multi authenticate homomorphic signature algorithm. In: *2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*. [S.l.: s.n.], 2023. p. 1–5. 3

VIJAYAKUMAR, V.; JOSEPH, K. S. Adaptive load balancing schema for efficient data dissemination in vehicular ad-hoc network vanet. *Alexandria Engineering Journal*, Elsevier, v. 58, n. 4, p. 1157–1166, 2019. 26, 27

WEBER, J. S.; FERRETO, T.; ZINCIR-HEYWOOD, N. Exploring anomaly detection techniques for enhancing vanet availability. In: *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*. [S.l.: s.n.], 2023. p. 1–7. 20, 24

WEISSMAN, S. A.; ANDERSON, N. G. Design of experiments (doe) and process optimization. a review of recent publications. *Organic Process Research & Development*, ACS Publications, v. 19, n. 11, p. 1605–1633, 2015. 57

WU, T.-Y.; OBAIDAT, M. S.; CHAN, H.-L. Qualityscan scheme for load balancing efficiency in vehicular ad hoc networks (vanets). *Journal of Systems and Software*, v. 104, p. 60–68, 2015. ISSN 0164-1212. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0164121215000321>. 26, 27

WU, X. et al. Mobility prediction-based joint task assignment and resource allocation in vehicular fog computing. In: IEEE. *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. [S.l.], 2020. p. 1–6. 20, 23

WU, Y. et al. Load balance guaranteed vehicle-to-vehicle computation offloading for min-max fairness in vanets. *IEEE Transactions on Intelligent Transportation Systems*, v. 23, n. 8, p. 11994–12013, 2022. 26, 27

YU, H. et al. An rsu deployment strategy based on traffic demand in vehicular ad hoc networks (vanets). *IEEE Internet of Things Journal*, IEEE, v. 9, n. 9, p. 6496–6505, 2021. 11, 12

ZHANG, H.; WEI, L. Topology characteristic analysis of vehicular ad hoc network based on time-varying complex network. *AIP Advances*, AIP Publishing, v. 11, n. 11, 2021. 11

ZHANG, X. et al. Faster parking and less cruise for public parking spot discovery: Modeling and analysis based on timed petri nets. In: IEEE. *2016 IEEE 13th International Conference on Networking, Sensing, and Control (ICNSC)*. [S.l.], 2016. p. 1–6. 5

Anexos

ANEXO A – DoE factor combination tables

Tabela 14 – Combination of factors considering the Availability metric.

EDGE_F	NET_F	RSU_F	RSU_R	LOG_F	Availability(%)
125.89	83220.00	500.00	2.00	168.00	98.35
125.89	83220.00	500.00	2.00	210.00	97.29
125.89	83220.00	500.00	3.00	168.00	97.76
125.89	83220.00	500.00	3.00	210.00	97.41
125.89	83220.00	750.00	2.00	168.00	98.80
125.89	83220.00	750.00	2.00	210.00	98.44
125.89	83220.00	750.00	3.00	168.00	97.93
125.89	83220.00	750.00	3.00	210.00	98.20
125.89	104025.00	500.00	2.00	168.00	97.82
125.89	104025.00	500.00	2.00	210.00	98.02
125.89	104025.00	500.00	3.00	168.00	98.09
125.89	104025.00	500.00	3.00	210.00	97.77
125.89	104025.00	750.00	2.00	168.00	98.62
125.89	104025.00	750.00	2.00	210.00	98.69
125.89	104025.00	750.00	3.00	168.00	98.04
125.89	104025.00	750.00	3.00	210.00	98.61
157.36	83220.00	500.00	2.00	168.00	97.80
157.36	83220.00	500.00	2.00	210.00	98.09
157.36	83220.00	500.00	3.00	168.00	97.16
157.36	83220.00	500.00	3.00	210.00	97.30
157.36	83220.00	750.00	2.00	168.00	98.50
157.36	83220.00	750.00	2.00	210.00	98.71
157.36	83220.00	750.00	3.00	168.00	98.07
157.36	83220.00	750.00	3.00	210.00	98.50
157.36	104025.00	500.00	2.00	168.00	98.16
157.36	104025.00	500.00	2.00	210.00	98.02
157.36	104025.00	500.00	3.00	168.00	97.01
157.36	104025.00	500.00	3.00	210.00	97.92
157.36	104025.00	750.00	2.00	168.00	99.02
157.36	104025.00	750.00	2.00	210.00	98.62
157.36	104025.00	750.00	3.00	168.00	98.06
157.36	104025.00	750.00	3.00	210.00	98.16

Tabela 15 – Combination of factors considering the MRT (ms) metric.

N_VM	VM_RES	Weight_Reins	Weight_Fail	QS	MRT (ms)
4.00	4.00	0.10	0.10	100.00	108166.63
4.00	4.00	0.10	0.10	1000.00	27610.26
4.00	4.00	0.10	0.29	100.00	52832.97
4.00	4.00	0.10	0.29	1000.00	34846.09
4.00	4.00	0.29	0.10	100.00	91579.99
4.00	4.00	0.29	0.10	1000.00	35804.40
4.00	4.00	0.29	0.29	100.00	60670.22
4.00	4.00	0.29	0.29	1000.00	32859.73
4.00	48.00	0.10	0.10	100.00	905.55
4.00	48.00	0.10	0.10	1000.00	731.01
4.00	48.00	0.10	0.29	100.00	667.45
4.00	48.00	0.10	0.29	1000.00	770.05
4.00	48.00	0.29	0.10	100.00	669.81
4.00	48.00	0.29	0.10	1000.00	711.95
4.00	48.00	0.29	0.29	100.00	1029.06
4.00	48.00	0.29	0.29	1000.00	711.40
8.00	4.00	0.10	0.10	100.00	3203.23
8.00	4.00	0.10	0.10	1000.00	2778.03
8.00	4.00	0.10	0.29	100.00	2157.43
8.00	4.00	0.10	0.29	1000.00	3553.21
8.00	4.00	0.29	0.10	100.00	3030.04
8.00	4.00	0.29	0.10	1000.00	2775.85
8.00	4.00	0.29	0.29	100.00	2799.20
8.00	4.00	0.29	0.29	1000.00	2807.41
8.00	48.00	0.10	0.10	100.00	805.78
8.00	48.00	0.10	0.10	1000.00	777.56
8.00	48.00	0.10	0.29	100.00	798.37
8.00	48.00	0.10	0.29	1000.00	754.15
8.00	48.00	0.29	0.10	100.00	811.24
8.00	48.00	0.29	0.10	1000.00	796.29
8.00	48.00	0.29	0.29	100.00	737.63
8.00	48.00	0.29	0.29	1000.00	824.75

Tabela 16 – Combination of factors considering the energy consumption metric.

N_C	**C_R**	**P_R**	**P_F**	**Q_S**	**EC (kW/h)**
4.00	4.00	0.10	0.10	100.00	0.49
4.00	4.00	0.10	0.10	1000.00	0.52
4.00	4.00	0.10	0.29	100.00	0.51
4.00	4.00	0.10	0.29	1000.00	0.51
4.00	4.00	0.29	0.10	100.00	0.57
4.00	4.00	0.29	0.10	1000.00	0.52
4.00	4.00	0.29	0.29	100.00	0.55
4.00	4.00	0.29	0.29	1000.00	0.53
4.00	48.00	0.10	0.10	100.00	0.53
4.00	48.00	0.10	0.10	1000.00	0.60
4.00	48.00	0.10	0.29	100.00	0.51
4.00	48.00	0.10	0.29	1000.00	0.53
4.00	48.00	0.29	0.10	100.00	0.48
4.00	48.00	0.29	0.10	1000.00	0.56
4.00	48.00	0.29	0.29	100.00	0.54
4.00	48.00	0.29	0.29	1000.00	0.47
8.00	4.00	0.10	0.10	100.00	0.54
8.00	4.00	0.10	0.10	1000.00	0.52
8.00	4.00	0.10	0.29	100.00	0.51
8.00	4.00	0.10	0.29	1000.00	0.51
8.00	4.00	0.29	0.10	100.00	0.52
8.00	4.00	0.29	0.10	1000.00	0.50
8.00	4.00	0.29	0.29	100.00	0.51
8.00	4.00	0.29	0.29	1000.00	0.50
8.00	48.00	0.10	0.10	100.00	0.58
8.00	48.00	0.10	0.10	1000.00	0.60
8.00	48.00	0.10	0.29	100.00	0.58
8.00	48.00	0.10	0.29	1000.00	0.52
8.00	48.00	0.29	0.10	100.00	0.51
8.00	48.00	0.29	0.10	1000.00	0.61
8.00	48.00	0.29	0.29	100.00	0.49
8.00	48.00	0.29	0.29	1000.00	0.54