Avaliação de Desempenho de Redes Veiculares Ad Hoc com RSUs e Filas de Prioridade

Jorge Rafael Loiola de Macêdo[1], **Vandirleya Barbosa**[1], **Leonel Feitosa Correia** [1]
Geraldo Pereira Rocha Filho [2], **Rodolfo Ipolito Meneguette** [3],
Luiz Nelson dos Santos Lima [1], **Francisco Airton Silva**[1]

[1] Universidade Federal do Piauí (UFPI)

[2]Universidade Estadual do Sudoeste da Bahia (UESB)

[3]Universidade de São Paulo (USP)

{jorge.rafael,vandirleya.barbosa,leonelfeitosa,luizznelson}@ufpi.edu.br

geraldo.rocha@uesb.edu.br,meneguette@icmc.usp.br,faps@ufpi.edu.br

Resumo. *O crescimento da frota de veículos impõe desafios significativos à organização e fluidez do tráfego, especialmente em cenários imprevisíveis, como emergências médicas e acidentes graves. Nesses contextos, o gerenciamento eficiente de requisições com diferentes níveis de prioridade é essencial para minimizar riscos e garantir o atendimento adequado a mensagens críticas. Este trabalho apresenta uma análise de desempenho de VANETs considerando múltiplas filas de prioridade e escalonamento circular. Utilizando Redes de Petri Estocásticas, o estudo modela o comportamento de RSUs e servidores Fog sob diferentes cargas de trabalho, avaliando métricas como tempo de resposta, utilização de recursos e probabilidade de descarte. Os resultados demonstram que o escalonamento circular melhora a distribuição de recursos, reduzindo em a probabilidade de descarte de mensagens críticas e garantindo melhor tempo médio de resposta para requisições prioritárias. Além disso, a configuração proposta permite uma utilização mais equilibrada da Fog Computing, evitando subutilização dos servidores e otimizando a alocação de processamento. O modelo fornece diretrizes para a otimização da infraestrutura VANET, auxiliando no dimensionamento de recursos e na melhoria da eficiência e confiabilidade dos sistemas de tráfego inteligente.*

Abstract. *The growth of the vehicle fleet poses significant challenges to the organization and fluidity of traffic, especially in unpredictable scenarios, such as medical emergencies and serious accidents. In these contexts, efficient management of requests with different priority levels is essential to minimize risks and ensure adequate service to critical messages. This work presents a performance analysis of VANETs considering multiple priority queues and circular scheduling. Using Stochastic Petri Nets, the study models the behavior of RSUs and Fog servers under different workloads, evaluating metrics such as response time, resource utilization, and discard probability. The results demonstrate that circular scheduling improves resource distribution, reducing the probability of discarding critical messages and ensuring better average response time for priority requests. In addition, the proposed configuration allows a more balanced*

use of Fog Computing, avoiding underutilization of servers and optimizing processing allocation. The model provides guidelines for optimizing VANET infrastructure, helping in resource sizing and improving the efficiency and reliability of intelligent traffic systems.

1. Introdução

Em 2024 foram vendidos 1.948.136 carros novos no Brasil, representando aumento de 13,21% em relação ao ano de 2023 [fenabrave 2025]. Estimativas apontam que em 2030 haverá cerca de 20,8 milhões de veículos autônomos só nos Estados Unidos [WIRE 2018]. Em dezembro de 2024, o Brasil possuía uma frota total de 122,9 milhões de veículos, com 85,8 milhões em circulação [RENASET 2024]. No mesmo ano foram registrados 858.622 acidentes que ocasionaram 13.225 óbitos [RENASET 2024]. A necessidade de sistemas inteligentes para organizar o tráfego e mitigar esses impactos é cada vez mais evidente. Informações sobre as condições da via melhoram a segurança e eficiência dos sistemas de transporte [Sinha and Mishra 2014], bem como diminuem as taxas de acidentes [Ismail et al. 2022].

A mobilidade urbana é dinâmica e exige respostas rápidas para otimização do fluxo de veículos. Tecnologias de Sistemas Inteligentes de Transporte (ITS) permitem a comunicação entre veículos e infraestrutura, fornecendo dados para um melhor controle do tráfego e aprimorando a segurança viária [Ismail et al. 2022, Sinha and Mishra 2014]. As Redes Veiculares Ad Hoc (VANETs) são redes sem fio descentralizadas e auto-organizadas, onde os veículos trocam informações relevantes entre si e com a infraestrutura ao longo da via [Mahmood and Horváth 2020, Balzano and Stranieri 2019, Yu et al. 2021, Shahin et al. 2023]. Nesse contexto, as Redes Veiculares Ad Hoc (VANETs) desempenham um papel crucial ao possibilitar a troca eficiente de informações em tempo real, permitindo desde alertas de congestionamento até comunicação entre veículos autônomos. As *Roadside Units* (RSUs) são equipamentos instalados ao longo das vias e interseções [Hota et al. 2022], que processam em tempo real as informações enviadas pelos veículos e por outras unidades antes de enviá-los para a *cloud* [Shahin et al. 2023]. As informações processadas são encaminhadas para os veículos próximos, outras RSUs e demais componentes do ITS. Esse modo de comunicação se refere a *Vehicle to Infrastructure* (V2I), que geralmente abrange curta distância entre um veículo e um nó da infraestrutura [Lim et al. 2021]. Os veículos são equipados com *On-board Units* (OBUs) para se comunicarem com as RSUs, ou com outros veículos, *Vehicle to Vehicle* (V2V) [Hota et al. 2022].

O crescimento da frota aumenta a troca de dados entre os veículos e demais entidades da VANET [Shahin et al. 2023], assim a velocidade do transporte de dados está se tornando o gargalo para computação baseada em *cloud* [Shi et al. 2016]. A computação em *edge* se refere às tecnologias que permitem que a computação seja realizada na borda da rede, para que ela aconteça perto das fontes de dados. Dispositivos como RSUs podem atuar como nós de borda, processando dados localmente e reduzindo a dependência da *cloud* para tomadas de decisão em tempo real. A Computação em *fog* se refere a um paradigma de computação distribuída que move o armazenamento e processamento para perto dos nós finais da rede com o objetivo de reduzir a sobrecarga da rede e computar as informações coletadas o mais rápido possível [Bellavista et al. 2019]. Aplicações que

exigem computação com *fog* são veículos conectados, veículos autônomos, redes inteligentes, redes de sensores e atuadores sem fio, casas inteligentes, cidades inteligentes e sistemas móveis de saúde [Chen et al. 2017]. Algumas dessas aplicações fazem parte dos ITS e exigem processamento em tempo real, devido ao volume de dados gerados e à mobilidade dos nós (veículos) [Shahin et al. 2023]. Desta forma, a conectividade entre veículos e infraestrutura se mostram aliadas à segurança, condução e otimização do fluxo de trânsito. Ambientes urbanos de alta complexidade requerem gerenciamento de cenários de tráfego denso. Os cenários que geram riscos à vida são os mais críticos, tais como desastres naturais, acidentes ou emergências médicas. A capacidade de reorganizar o tráfego em tempo real e identificar rotas alternativas impactam na eficiência das operações de resgate. A segurança pública também é beneficiada, além de contribuir para redução de acidentes. Neste contexto, é importante garantir que veículos de emergência recebam a devida prioridade.

RSUs demonstram frequentemente ganhos significativos em termos de latência e qualidade de serviço, especialmente em redes urbanas e rodoviárias com alta demanda de comunicação [Mahmood and Horváth 2020, Nidhi and Lobiyal 2021]. Veículos e infraestrutura trocam informações com diferentes níveis de prioridade em uma VANET. Gerenciar múltiplas requisições com diferentes níveis de prioridade é essencial para minimizar o risco à vida, exigindo soluções que priorizem mensagens críticas. Para lidar com esses desafios, este estudo propõe um modelo baseado em múltiplas filas de prioridade e escalonamento circular, otimizando o fluxo de mensagens críticas e reduzindo descartes. A solução utiliza Redes de Petri Estocásticas (SPN) para modelar o comportamento das RSUs e servidores Fog sob diferentes cargas de trabalho, permitindo prever e melhorar o desempenho das VANETs. Assim, podemos resumir as principais contribuições desta pesquisa da seguinte forma: ***(i)*** Propor um modelo SPN para simular o comportamento de filas prioritárias em VANETs; ***(ii)*** Analisar como diferentes configurações de filas de prioridade afetam o desempenho de RSUs na camada *Edge*, bem como servidores na camada *Fog*, para identificar gargalos e otimizar recursos em situações de alta demanda; ***(iii)*** Avaliar o impacto de grandes volumes de requisições sobre o tempo de resposta, probabilidade de descarte, utilização das camadas *Edge* e *Fog*.

O restante deste estudo está estruturado da seguinte maneira: A Seção 2 expõe pesquisas relacionadas por meio de uma tabela comparativa entre os estudos avaliados. A Seção 3 descreve a arquitetura adotada neste estudo. A Seção 4 apresenta o modelo SPN sugerido e as métricas empregadas para a análise de desempenho. Na Seção 5, os resultados apresentados e detalhados. A Seção 6 apresenta as conclusões obtidas e discute pesquisas futuras.

2. Trabalhos Relacionados

Esta seção apresenta os trabalhos relacionados com abordagens ou contextos a este estudo. A seleção de trabalhos considerou critérios como Tipo de Modelo, Métricas, uso de RSUs, Múltiplas Filas de Prioridade, uso de Escalonamento e Foco. A Tabela 1 agrupa as contribuições dos trabalhos com maior relevância ao tema deste estudo. Para organizar as abordagens da literatura, os trabalhos foram agrupados em categorias temáticas com base em seus principais objetivos.

Os trabalhos focados em protocolos de roteamento analisam como otimizar a

Tabela 1. Comparação de abordagens de aprendizagem federada

Trabalho	Tipo de Modelo	Métricas	Usa RSUs	Múltiplas Filas de Prioridade	Usa Escalonamento	Foco
[Hota et al. 2022]	FRIIS, Two-Ray Ground, Log-Distance, Nakagami	Throughput, taxa de entrega de pacotes, atraso end-to-end, goodput médio, overhead	✓	✗	✗	Protocolos de Roteamento
[Mahmood and Horváth 2020]	Markov Chain	Velocidade de propagação de mensagens, distribuição transitória da distância	✓	✗	✗	Infraestrutura e Gerenciamento
[Wang et al. 2023]	Confiança distribuída	Precisão, recall, taxa de entrega de mensagens	✓	✗	✗	Segurança e Veículos Maliciosos
[Mousa et al. 2021]	Matemático	Taxa de conectividade	✓	✗	✗	Conectividade
[Shahin et al. 2023]	Matemático	Cobertura de estrada, demanda computacional satisfeita	✓	✗	✗	Conectividade
[Ghosh et al. 2023]	IIA-ORD	Cobertura média, razão de tempo de cobertura, taxa de entrega de pacotes, atraso fim-a-fim	✓	✗	✗	Infraestrutura e Gerenciamento
[Nidhi and Lobiyal 2021]	Car-Following, Two-Ray Ground	Packet Delivery Ratio, Packet Loss, Routing Overhead, End-to-End Delay	✓	✗	✗	Infraestrutura e Gerenciamento
[Rodrigues et al. 2021b]	SPN	MRT, probabilidade de descartes, número de descartes, utilização dos RSUs	✓	✗	✗	Infraestrutura e Gerenciamento
[Ullah et al. 2023]	EMR-ICN	Taxa de entrega de mensagens, latência, número médio de saltos	✓	✗	✗	Protocolos de Roteamento
[Tan and Chung 2021]	Matemático	Consumo de armazenamento, tempo durante o processo de autenticação das RSUs	✓	✗	✗	Segurança e Veículos Maliciosos
[Atwa et al. 2021]	RTEAM	Validação de eventos, detecção de eventos, confiabilidade, estimativa de risco	✓	✗	✗	Segurança e Veículos Maliciosos
[Gao et al. 2020]	Matemático	Taxa de entrega de pacotes, atraso de tempo, saltos sem fio	✓	✗	✗	Protocolos de Roteamento
Este trabalho	SPN	MRT, probabilidade de descartes, utilização	✓	✓	✓	Infraestrutura e Gerenciamento

disseminação de pacotes em VANETs e melhorar a comunicação entre veículos e infraestruturas. O trabalho [Hota et al. 2022] apresenta uma análise comparativa de protocolos de roteamento e modelos de propagação para otimizar a disseminação de pacotes em VANETs. O [Ullah et al. 2023] propõe um esquema de seleção de relé confiável para o roteamento de mensagens de emergência, utilizando comunicação híbrida V2V e V2I, além de predição de posição e métricas de mobilidade para garantir a estabilidade da rota. Por fim, o [Gao et al. 2020] apresenta um esquema de roteamento híbrido que combina transmissões V2V e V2R, utilizando RSUs para aumentar a confiabilidade e o desempenho da rede.

Os estudos com foco em conectividade investigam maneiras de melhorar a conectividade entre veículos e infraestrutura. O [Rodrigues et al. 2021b] propõe um modelo para avaliar o desempenho de redes veiculares que utilizam V2V e V2I, destacando os benefícios do *multihoming* em rodovias *multilane* para melhorar a conectividade e a estabilidade da taxa de dados. O [Mousa et al. 2021] propõe um modelo para determinar as melhores posições para implantar RSUs em rodovias, visando maximizar a conectividade

da rede. O [Shahin et al. 2023] sugere uma estratégia baseada em *fog computing* para otimizar a configuração e localização de RSUs, utilizando técnicas de Teorias de Módulos de Satisfatibilidade (SMT) para minimizar custos e maximizar a qualidade de serviço.

Os trabalhos com foco em segurança e veículos maliciosos discutem como lidar com ameaças e proteger a comunicação nas redes veiculares. O [Wang et al. 2023] apresenta um modelo de confiança distribuída que inclui um mecanismo de apelação contra punições injustas por RSUs comprometidas, melhorando a precisão e confiabilidade na detecção de veículos maliciosos. O [Tan and Chung 2021] propõe um design de associação de grupo de Veículos Aéreos Não Tripulados (UAVs) para disseminação de mensagens V2V em ambientes remotos, aprimorando a conectividade e segurança. Por fim, o [Atwa et al. 2021] apresenta o modelo RTEAM, que utiliza confiança baseada em risco para tomar decisões sobre a validade de eventos em VANETs, considerando a autenticidade do remetente e o risco associado a cada ação.

Trabalhos com foco em infraestrutura e gerenciamento tratam de soluções para otimizar a implantação e operação de RSUs e o gerenciamento de redes veiculares. O [Mahmood and Horváth 2020] apresenta um modelo para caracterizar o processo de propagação de mensagens em VANETs, determinando a velocidade de propagação e a distribuição transitória da distância onde a mensagem está disponível em uma infraestrutura. O [Ghosh et al. 2023] propõe um sistema para otimizar a implantação de RSUs em interseções influentes, utilizando um modelo baseado em K-shell modificado e Técnica para a Ordem de Preferência por Similaridade à Solução Ideal (TOPSIS), validado por um modelo de previsão de tráfego baseado em Memória de Longo e Curto Prazo (LSTM) bidirecional empilhado. O [Nidhi and Lobiyal 2021] sugere uma estratégia para implantar RSUs em áreas urbanas densas, visando melhorar a comunicação V2I. O [Rodrigues et al. 2021b] propõe um modelo SPN para analisar o desempenho de uma infraestrutura VANET com múltiplos semáforos inteligentes cooperativos, avaliando métricas como tempo médio de resposta, utilização de recursos e número de descartes de requisições.

Diante dos trabalhos supracitados, observa-se que as abordagens existentes apresentam limitações que dificultam a priorização de mensagens críticas em VANETs. Modelos baseados apenas em políticas estáticas de priorização não conseguem se adaptar a cenários dinâmicos, enquanto soluções baseadas em aprendizado de máquina apresentam alta demanda computacional e dificuldades na obtenção de dados rotulados de qualidade. Diante desses desafios, este trabalho propõe uma abordagem híbrida baseada em SPN e escalonamento dinâmico de mensagens, otimizando a alocação de prioridade em VANETs sem comprometer a escalabilidade. Diferentemente dos modelos tradicionais, nossa proposta permite que mensagens críticas sejam processadas com maior eficiência, reduzindo atrasos e evitando congestionamentos que possam comprometer a segurança do tráfego. Os resultados de desempenho encontrados servem como *input* para trabalhos que tenham foco em protocolos de roteamento, conectividade, segurança da comunicação, QoS, dentre outros contextos na temática de VANETs.

3. Arquitetura

Esta seção apresenta uma arquitetura VANET com RSU, que servirá de referência para a modelagem e análise de desempenho. A Figura 1 representa a arquitetura proposta, que é

dividida em três camadas chamadas *(i)* Admissão, *(ii) Edge* e *(iii) Fog*. A camada de Admissão representa os veículos ao longo da via, que enviam requisições de diversos níveis de prioridade. A segunda camada possui RSUs, que recebem as mensagens dos veículos próximos, processam e encaminham para camada seguinte. O servidor na camada de *Fog* recebe as requisições e realiza o processamento final. Para fins deste estudo, a arquitetura abstrai a comunicação V2V, bem como da infraestrutura para os veículos, além de valiar a interação de uma RSU com o servidor de *Fog*.

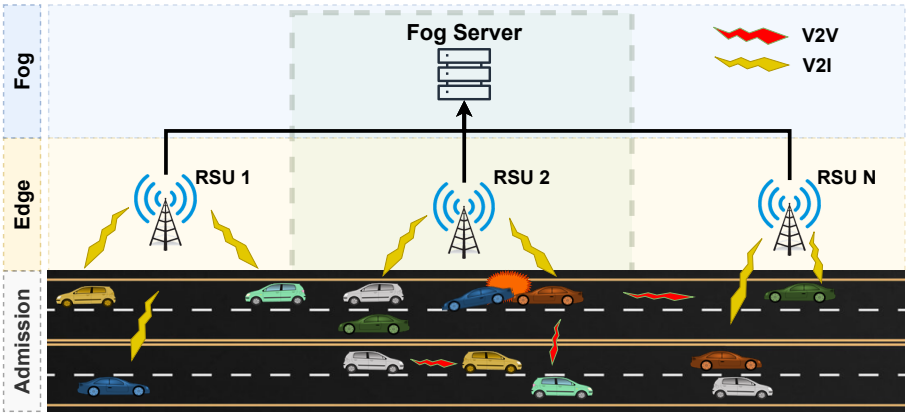


Figura 1. Arquitetura proposta

A Comissão Federal de Comunicação dos Estados Unidos (FCC) licenciou a banda de frequência de 5,850–5,925 GHz para comunicação veicular, com sete canais separados [Hota et al. 2022, Ismail et al. 2022]. Um canal é dedicado para transmitir dados de controle e os demais canais são habilitados para requisições de outros serviços, seja de entretenimento, seja de segurança e emergência [Ismail et al. 2022]. Para simplificar, a arquitetura considerou apenas três canais de comunicação. As requisições que trafegavam pelos canais foram agrupadas em três classes de prioridade chamadas `C1`, `C2` e `C3`. As requisições de classe `C1` possuem maior prioridade e abrangem mensagens como alertas de desastres naturais, evacuação e sincronização entre os componentes da VANET. As requisições de classe `C2` possuem média prioridade e agrupam mensagens de alerta de colisão, alerta de passagem de veículos dos serviços de emergência e segurança. As requisições de classe `C3` possuem menor prioridade e englobam os demais tipos de requisições relacionadas a congestionamento, conforto e entretenimento dos passageiros.

Considerando princípios de Teoria das Filas, em uma arquitetura sem distinção de canais, as requisições de todas as classes de prioridade ocupam a mesma fila de processamento, que geralmente segue uma disciplina como *First Come, First Served* (FCFS). Caso existam muitas requisições de baixa prioridade, as requisições de alta e média prioridade podem demorar a serem processadas. A arquitetura proposta adota três filas de prioridade, que receberão as requisições de classe `C1`, `C2` e `C3`, correspondentes aos três canais de comunicação. Na camada *Edge*, a RSU classifica cada requisição e colocada na fila de prioridade apropriada. As requisições mantém a mesma classificação ao serem encaminhadas para a camada de *Fog*. A arquitetura contempla escalonamento circular entre as classes de prioridade. Esta abordagem evita que requisições de alta e média prioridade percam o objetivo, pois o tempo de espera para processamento será menor. Assim, cada classe terá determinada quantidade de requisições processadas, iniciando com a classe de

maior prioridade até a classe de menor prioridade, quando recomeça o ciclo.

Esta arquitetura considera um sistema de filas não preemptivo, ou seja, uma mensagem de maior prioridade nunca interrompe um processamento iniciado. O modelo não considera o tempo de vida de uma requisição. Neste caso uma requisição aceita continuará na fila até ser processada. Pontos de acesso possuem configurações homogêneas. Mecanismos de transmissão e roteamento entre RSUs não são necessários para este estudo.

4. Modelo SPN

Esta Seção descreve o modelo proposto, organizado em cinco componentes que representam as operações do sistema: *(1)* Admissão, *(2)* Unidade RSU, *(3)* Escalonador da RSU (R1), *(4)* Servidor de *Fog*, e *(5)* Escalonador do Servidor de *Fog* (R2). Os componentes foram elaborados para avaliar o desempenho do sistema em relação a demanda de tráfego e capacidade computacional, respeitando os parâmetros da arquitetura adotada. *Rede de Petri* (PN) é uma ferramenta de modelagem gráfica e matemática, que permite descrever e estudar sistemas de processamento de informações caracterizados como sendo concorrentes, assíncronos, distribuídos e paralelos [Lima et al. 2021].

4.1. Estrutura do Modelo

A Figura 2 representa o modelo SPN, para a arquitetura proposta na Figura 1. O módulo de admissão controla o fluxo de chegada das requisições de diferentes classes de prioridade i, onde $i \in \{1, 2, 3\}$. A transição temporizada $\texttt{ArrivalC}_\texttt{i}$ representa a geração de requisições para a classe $\texttt{C}_\texttt{i}$. Uma requisição (*token*) que chega para entrar no sistema ocupa o lugar $\texttt{ArrivedQC}_\texttt{i}$ correspondente à classe $\texttt{C}_\texttt{i}$. Novas requisições são admitidas conforme a disponibilidade de vagas no lugar `RQAvaliable` e alocadas na fila de prioridade apropriada, que é representada por $\texttt{RQC}_\texttt{i}$. O lugar `RPAvailable` representa a capacidade de processamento, que pode ter até `RPCapacity` núcleos disponíveis. Uma requisição em $\texttt{RQC}_\texttt{i}$ será processada se houver núcleo (*token*) disponível em `RPAvailable`. O lugar $\texttt{RPC}_\texttt{i}$ representa as requisições da classe $\texttt{C}_\texttt{i}$ em processamento. O disparo da transição temporizada $\texttt{TR}_\texttt{i}$ significa que uma requisição da classe $\texttt{C}_\texttt{i}$ foi processada.

Os dados processados na RSU são enviados para o servidor na camada de *Fog*, através de um *gateway*. De forma similar à RSU, as requisições são alocadas nas filas correspondentes do servidor de *Fog*. O lugar $\texttt{FSQC}_\texttt{i}$ representa uma fila de prioridade. O lugar `FSQAvailable` indica a disponibilidade de vagas no servidor e o lugar `FSPAvailable` representa a capacidade de processamento disponível. O lugar $\texttt{FSPC}_\texttt{i}$ indica as requisições de classe $\texttt{C}_\texttt{i}$ em processamento. O disparo da transição temporizada $\texttt{TFS}_\texttt{i}$ indica que uma requisição foi processada. A migração de *tokens* entre os lugares $\texttt{ArrivalC}_\texttt{i}$, $\texttt{RQC}_\texttt{i}$, $\texttt{RPC}_\texttt{i}$, $\texttt{FSQC}_\texttt{i}$ e $\texttt{FSPC}_\texttt{i}$ representa o fluxo de uma requisição de classe $\texttt{C}_\texttt{i}$ dentro do sistema.

No componente escalonador da RSU, o *token* no lugar `R1C1N` indica que requisições da classe `C1` serão processadas. Caso existam requisições em `RQC1`, então `R1C1D` receberá `R1Seq1` *tokens*, que representam a quantidade máxima de requisições da classe `C1` para processar em um ciclo. Quando `R1C1D` esvaziar ou não houverem mais requisições em `RQC1`, então o lugar `R1C2N` receberá um *token* indicando que requisições

da classe C2 serão processadas. Se existirem requisições a serem processadas em RQC2, então o lugar R1C2D receberá R1Seq2 *tokens*. O lugar R1C3N receberá o *token* de habilitação quando acaberem os *tokens* de R1C2D ou não houverem requisições em RQC2. O processo se repete para a classe C3 e retorna para a classe C1. O escalonador da máquina de *Fog* funciona de forma análoga. Cada máquina possui sua própria rotina de escalonamento devido às configurações de cada uma, bem como o volume de requisições dadas as condições de trânsito. Realizando uma análise qualitativa, o modelo possui as condições de guarda que evitam o estado *deadlock*.

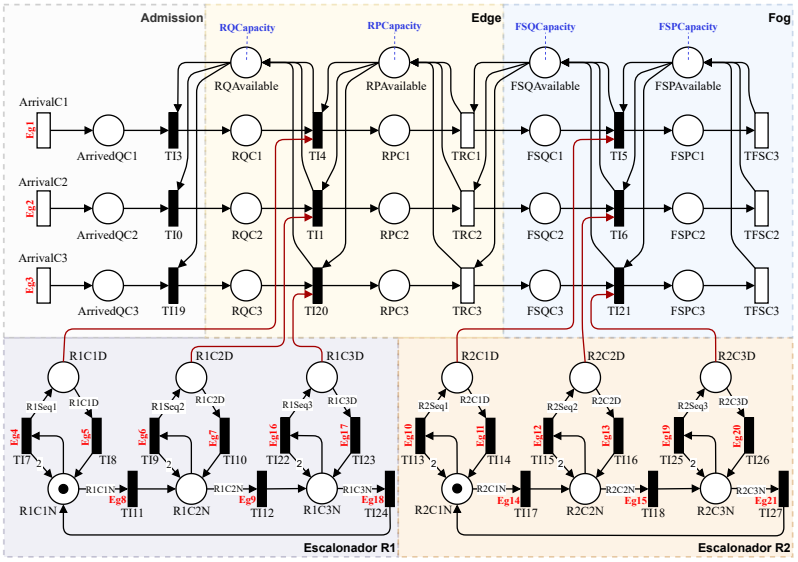


Figura 2. Modelo SPN contendo três filas de prioridades, usando escalonamento entre as filas

4.2. Métricas e Condições de Guarda

As métricas adotadas nesse trabalho são: Tempo Médio de Resposta (MRT), Utilização da RSU (URSU), Utilização do Servidor de *Fog* (UFS), Utilização da Fila na RSU (UQRSU), Utilização da Fila no Servidor de *Fog* (UQFS) e Probabilidade de Descarte (DP). O cálculo do MRT é efetuado pelo total de requisições dentro do sistema multiplicado pelo Atraso de Chegada (AD), conforme a Lei de Little [Little 1961]. Um sistema estável é pré-requisito para a Lei de Little, ou seja, que a taxa de chegadas seja menor do que a taxa de serviço. Desta forma o sistema não ficará sobrecarregado. A Taxa de Chegada (AR) mede a frequência de mensagens que chegam ao sistema por unidade de tempo. A AR é calculada como o inverso do AD, que representa o tempo médio entre as chegadas consecutivas de mensagens. No contexto deste trabalho a AR é expressa em milissegundos (ms).

MRT é igual ao produto da quantidade de requisições em progresso, multiplicado pelo tempo entre chegadas de requisições, no entanto, o modelo proposto possui três entradas de dados simultâneas. Assim, para calcular o MRT no modelo proposto foi necessário utilizar o tempo da última transição, denominada $\texttt{FSST}_i$, ou seja, o tempo médio de serviço do servidor da camada de *Fog*. Para encontrar o número de requisições em progresso, é necessário obter a quantidade momentânea total de *tokens* que estão sendo processados desde o início do modelo até o final. Para esse cálculo basta somar

a esperança estatística de existir *tokens* em todos os lugares onde a requisição passará [Rodrigues et al. 2021a]. Por não utilizarmos o tempo médio entre chegadas, o cálculo do MRT deverá ser multiplicado pela probabilidade de se ter ao menos um *token* no lugar $\texttt{FSPC}_\texttt{i}$.

A métrica URSU calcula a porcentagem da capacidade processamento da RSU, que está sendo consumida a uma uma determinada taxa de chegada de requisições. A métrica UFS, se refere a utilização da capacidade de processamento do servidor da *Fog*. A métrica UQRSU está associada à utilização da capacidade de fila na RSU, ou seja, a ocupação da fila por requisições aguardando processamento. De forma análoga, a métrica UQFS analisa a utilização da fila no servidor da *Fog*. Já a DP se refere às requisições que chegam ao sistema, mas que podem ser descartadas devido a elevada ocupação do sistema causada por sobrecarga na utilização. O descarte pela camada *Fog* não é abordado, pois o modelo considera que uma requisição admitida será processada. Para calcular o comportamento das filas de prioridade, o modelo utiliza as métricas listadas na Tabela 2 e as condições de guarda listadas na Tabela 3. O modelo SPN foi projetado na ferramenta *Mercury* [Maciel et al. 2017] (versão 5.0.2) O tempo médio de serviço das transições $\texttt{TR}_\texttt{i}$ e $\texttt{TFS}_\texttt{i}$ foram extraídos da literatura sobre o tema [Silva et al. 2024]. Os demais parâmetros estão elencados na Tabela 4.

Tabela 2. Métricas utilizadas no modelo SPN

Métrica	Expressão	Descrição
DPC_i	$P\{(\#ArrivedQC_\text{i} > 0)AND(\#RQAvailable = 0)\} * 100$	DP da classe C_i, onde $i \in \{1, 2, 3\}$
DPTotal	$(DPC1 + DPC2 + DPC3) * 100$	DP na entrada do sistema
MRTC_i	$(((E\{\#RQC_\text{i}\}) + (E\{\#RPC_\text{i}\}) + (E\{\#FSQC_\text{i}\}) + (E\{\#FSPC_\text{i}\})) * FSST) * P\{(\#FSPC_\text{i} > 0)\}$	MRT da classe C_i, onde $i \in \{1, 2, 3\}$
MRTTotal	$(MRTC1 + MRTC2 + MRTC3)$	MRT total do sistema
UQRC_i	$(E\{\#RQC_\text{i}\}/RQCapacity) * 100$	Uso da fila pela classe C_i na RSU
UQRTotal	$(((E\{\#RQC_1\}) + (E\{\#RQC_2\}) + (E\{\#RQC_3\}))/RQCapacity) * 100$	Uso total das filas na RSU
UQFSC_i	$(E\{\#FSQC_\text{i}\}/FSQCapacity) * 100$	Uso da fila pela classe C_i na *Fog*
UQR2Total	$(((E\{\#FSQC_1\}) + (E\{\#FSQC_2\}) + (E\{\#FSQC_3\}))/FSQCapacity) * 100$	Uso total das filas na *Fog*
URC_i	$(E\{\#RPC_\text{i}\}/RPCapacity) * 100$	Utilização da classe C_i na RSU
URTotal	$(((E\{\#RPC_1\}) + (E\{\#RPC_2\}) + (E\{\#RPC_3\}))/RPCapacity) * 100$	Utilização total na RSU
UFSC_i	$(E\{\#FSPC_\text{i}\}/FSPCapacity) * 100$	Utilização da classe C_i na *Fog*
UFSTotal	$(((E\{\#FSPC_1\}) + (E\{\#FSPC_2\}) + (E\{\#FSPC_3\}))/FSPCapacity) * 100$	Utilização total na *Fog*

5. Estudo de Caso

Esta seção apresenta os resultados obtidos, que foram discutidos como estudo de caso. A Figura 3(a) apresenta a relação entre AR e a taxa de utilização da RSU, que possui elevada taxa de utilização inicial e sofre saturação ainda com baixas taxas de chegada ($<$ 0,3 msg/ms) para a utilização total. Na mesma faixa, a avaliação por classes mostra que $\texttt{C1}$ rapidamente alcança 40% de utilização e domina progressivamente o uso do recurso, alcançando por volta de 65% de utilização com 1,2 msg/ms. As classes $\texttt{C2}$ e $\texttt{C3}$ possuem taxas de chegada constantes e apresentam maior utilização inicial, entretanto a sua utilização decresce a medida que a taxa de chegada da classe $\texttt{C1}$ aumenta.

Tabela 3. Condições de Guarda

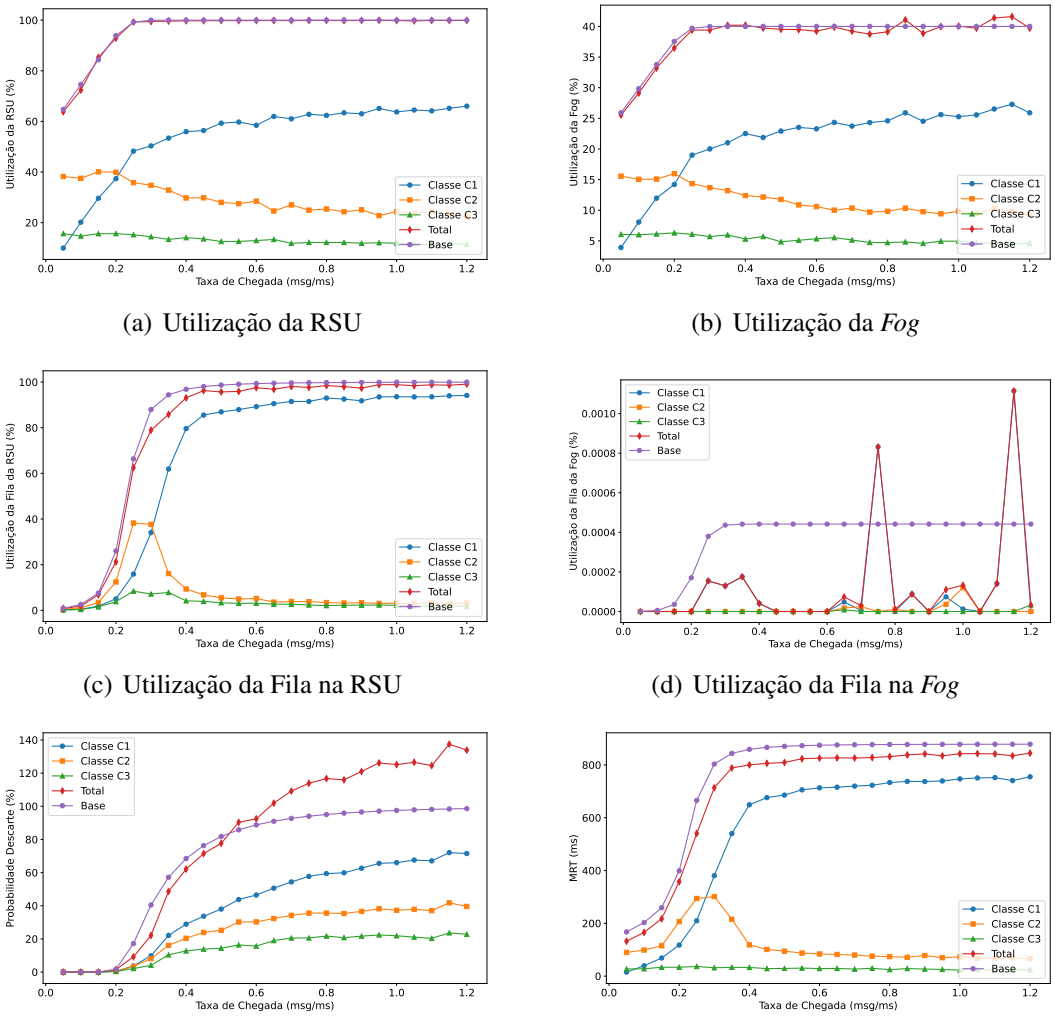
Índice da Expressão	Condição de Guarda
Eg1, Eg2, Eg3	$ArrivedQ <= 2$
Eg4	$(RQC1 > 0)AND(R1C1D = 0)AND(R1C1N = 1)$
Eg5	$(RQC1 = 0)AND(R1C1D > 0)$
Eg6	$(RQC2 > 0)AND(R1C2D = 0)AND(R1C2N = 1)$
Eg7	$(RQC2 = 0)AND(R1C2D > 0)$
Eg8	$((R1C1D = 0)AND(R1C1N >= 2))OR(((RQC1 = 0)AND(R1C1N = 1))AND((RQC2 > 0)OR(RQC3 > 0)))$
Eg9	$((R1C2D = 0)AND(R1C2N >= 2))OR(((RQC2 = 0)AND(R1C2N = 1))AND((RQC1 > 0)OR(RQC3 > 0)))$
Eg10	$(FSQC1 > 0)AND(R2C1D = 0)AND(R2C1N = 1)$
Eg11	$(FSQC1 = 0)AND(R2C1D > 0)$
Eg12	$(FSQC2 > 0)AND(R2C2D = 0)AND(R2C2N = 1)$
Eg13	$(FSQC2 = 0)AND(R2C2D > 0)$
Eg14	$((R2C1D = 0)AND(R2C1N >= 2))OR(((FSQC1 = 0)AND(R2C1N = 1))AND((FSQC2 > 0)OR(FSQC3 > 0)))$
Eg15	$((R2C2D = 0)AND(R2C2N >= 2))OR(((FSQC2 = 0)AND(R2C2N = 1))AND((FSQC1 > 0)OR(FSQC3 > 0)))$
Eg16	$(RQC3 > 0)AND(R1C3D = 0)AND(R1C3N = 1)$
Eg17	$(RQC3 = 0)AND(R1C3D > 0)$
Eg18	$((R1C3D = 0)AND(R1C3N >= 2))OR(((RQC3 = 0)AND(R1C3N = 1))AND((RQC1 > 0)OR(RQC2 > 0)))$
Eg19	$(FSQC3 > 0)AND(R2C3D = 0)AND(R2C3N = 1)$
Eg20	$(FSQC3 = 0)AND(R2C3D > 0)$
Eg21	$((R2C3D = 0)AND(R2C3N >= 2))OR(((FSQC3 = 0)AND(R2C3N = 1))AND((FSQC1 > 0)OR(FSQC2 > 0)))$

Tabela 4. Definições utilizadas no modelo SPN

Categoria	Elemento	Descrição	Valor
Variáveis	AR1	Taxa de Chegada de mensagens em msg/ms, com incrementos de 0,05	0,05 a 1,2
	ADC1	Tempo médio entre chegadas da classe C1	1/AR1 ms
	ADC2	Tempo médio entre chegadas da classe C2	5.0 ms
	ADC3	Tempo médio entre chegadas da classe C3	12.5 ms
	RST	Tempo médio de serviço da RSU	15.7 ms
	FSST	Tempo médio de serviço do Servidor de *Fog*	15.7 ms
	RQCapacity	Capacidade máxima da fila na RSU	40
	RPCapacity	Núcleos de processamento da RSU	8
	FSQCapacity	Capacidade máxima da fila do Servidor de *Fog*	40
	FSPCapacity	Núcleos de processamento do Servidor de *Fog*	20
	R1Seq1, R2Seq1	Requisições atendidas da classe C1 na RSU e na *Fog*, respectivamente	6
	R1Seq2, R2Seq2	Requisições atendidas da classe C2 na RSU e na *Fog*, respectivamente	4
	R1Seq3, R2Seq3	Requisições atendidas da classe C3 na RSU e na *Fog*, respectivamente	2
Transições	ArrivalC$_i$	Tempo associado às chegadas da classe C_i, onde $i \in \{1, 2, 3\}$	ADC$_i$
	TR1, TR2 e TR3	Tempo associado ao processamento de uma requisição na RSU	RST
	TFS1, TFS2 e TFS3	Tempo associado ao processamento de uma requisição na *Fog*	FSST

É possível verificar que as classes C1 e C2 competem pelo processamento, mas a estratégia de priorização e atendimento mínimo estabelecidas pelo escalonador garante que as classes de menor prioridade continuem sendo atendidas. Isto quer dizer que em baixas taxas de chegada da classe C1, as classes de menor prioridade conseguem maior acesso ao processamento, porque a classe C1 envia poucas requisições. A classe C3 possui taxa de chegada baixa e constante, mesmo disputando pouco recurso a estratégia do escalo-

nador garante utilização em torno de 10% a 15%. Conforme o gráfico da Figura 3(b), no servidor da *Fog* a utilização total é moderada e alcança cerca de 40%. Na avaliação por classes, o comportamento é similar ao da RSU, onde as classes `C2` e `C3` reduzem a utilização a medida que `C1` aumenta. Ressalta-se que a *Fog* é subutilizada por causa da rápida saturação da RSU, que possui poucos núcleos de processamento. O desbalanceamento transforma a RSU no gargalo do sistema. Todavia a configuração da *Fog* permite incluir mais RSUs no sistema, podendo representar um pequeno grupo de RSUs em uma via.



(a) Utilização da RSU

(b) Utilização da *Fog*

(c) Utilização da Fila na RSU

(d) Utilização da Fila na *Fog*

(e) Probabilidade de Descarte

(f) Tempo Médio de Resposta (MRT)

Figura 3. Resultados da análise de desempenho

O gráfico da Figura 3(c) mostra a utilização da fila na RSU, onde a utilização total cresce exponencialmente e entra na curva de saturação com AR menor do que 0,4 msg/ms. A alta ocupação da fila é compatível com o momento de saturação do processamento, ou seja, a elevada taxa de chegada faz com que as requisições de todas as classes se acumulem. A classe `C2` utiliza próximo de 40% da capacidade da fila devido a iniciar com maior taxa de chegada. Entretanto, a elevada AR da classe `C1` provoca o crescimento exponencial de utilização da fila, dando corpo a taxa de utilização total. Destaca-se que

uma máquina processará um número máximo de requisições de cada classe por ciclo de escalonamento. Assim, o número limitado de processamento por ciclo faz com que a classe `C1` acumule requisições em fila. A queda de utilização da classe `C2` se dá em razão da concorrência com a classe `C1`, mas também porque o escalonador garante que determinada quantidade de requisições seja processada. A fila da classe `C3` tem um crescimento incipiente, mas por ter baixa taxa de chegada o escalonamento evita acumulo de requisições. A Figura 3(d) mostra a utilização da fila para a *Fog*, que fica a baixo de 0,001%, pois o servidor de *Fog* processa rapidamente as requisições, como verificado no gráfico da Figura 3(b). Desta forma, a fila é praticamente inexistente e qualquer variação na chegada de requisições gera os picos observados na Figura 3(d).

Na Figura 3(e), a probabilidade de descarte total cresce significativamente com o aumento do AR. A classe `C1` sofre descartes consideráveis, mesmo sendo de alta prioridade. As classes de menor prioridade (`C2` e `C3`) possuem menor probabilidade de descarte, levando-se em consideração que possuem menores taxas de chegada. Observa-se também que o sistema começa a descartar mensagens significativamente após AR $\approx$ 0,3 msg/ms. Na Figura 3(f), o MRT apresenta um padrão de crescimento semelhante ao observado na utilização da fila na RSU para todas as classes, confirmando que MRT é proporcional ao tamanho das respectivas filas. Ou seja, quanto maior a fila, maior o tempo que a requisição passa no sistema. Nesse contexto, o valor mais alto do MRT observado para a classe `C1` reflete, sobretudo, a demanda constante e intensa, demonstrando que o sistema permanece robusto mesmo sob cargas extremas. O escalonamento evita cenários extremos de subutilização ou saturação, o que facilita decisões futuras quanto ao dimensionamento adequado da infraestrutura. Diante desses resultados, pode-se afirmar que a proposta é viável em cenários reais nos quais o processamento eficiente de mensagens críticas pode significar diretamente salvar vidas.

6. Conclusão

Dada a complexidade envolvida no processamento de múltiplas requisições prioritárias dentro da infraestrutura VANET, o modelo proposto com escalonamento oferece uma abordagem coesa para a avaliação de desempenho. Ao considerar três classes de requisições (`C1`, `C2` e `C3`) e suas respectivas filas de prioridade, o sistema é capaz de gerenciar adequadamente a alocação de recursos sem que haja preempção entre os processos em execução. Ao tempo que evita o bloqueio das classes de menor prioridade, bem como evita que classes de maior prioridade aguarde demasiadamente, garantindo que uma quantidade mínima de requisições seja atendida em cada classe. A arquitetura demonstra uma organização eficiente, garantindo que as camadas sejam capazes de lidar com o fluxo de mensagens mesmo em cenários de alta demanda, otimizando o tempo de resposta e minimizando a probabilidade de descarte. Com base nos experimentos realizados, é possível concluir que o balanceamento entre as capacidades das *Edge* e *Fog* tem papel central no desempenho geral, destacando a necessidade de uma configuração adequada dos núcleos de processamento.

Este estudo aponta que o escalonamento garante a distribuição da capacidade de processamento. Ressalta-se que a quantidade de requisições que cada classe processará (em um ciclo do escalonamento) deve ser ajustada para otimizar o balanceamento entre as classes. Os resultados obtidos com a análise do modelo base devem ser observados como parâmetro no planejamento de dimensionamento dos componentes reais. Portanto, este

trabalho pode auxiliar os administradores de sistemas a identificarem as configurações mais adequada em termos de custo e eficiência. Para trabalhos futuros serão exploradas técnicas de preempção na análise de desempenho, bem como avaliar como diversas RSUs são afetadas quando há sobrecarga do sistema por eventual queda de outras RSUs.

Referências

Atwa, R. J., Flocchini, P., and Nayak, A. (2021). Rteam: Risk-based trust evaluation advanced model for vanets. *IEEE Access*, 9:117772–117783.

Balzano, W. and Stranieri, S. (2019). 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) 33*, pages 851–859. Springer.

Bellavista, P., Berrocal, J., Corradi, A., Das, S. K., Foschini, L., and Zanni, A. (2019). A survey on fog computing for the internet of things. *Pervasive and mobile computing*, 52:71–99.

Chen, S., Zhang, T., and Shi, W. (2017). Fog computing. *IEEE Internet Computing*, 21(2):4–6.

fenabrave (2025). Dados de mercado fenabrave ed. 264 - emplacamentos em 2024.

Gao, H., Liu, C., Li, Y., and Yang, X. (2020). V2vr: reliable hybrid-network-oriented v2v data transmission and routing considering rsus and connectivity probability. *IEEE Transactions on Intelligent Transportation Systems*, 22(6):3533–3546.

Ghosh, S., Saha Misra, I., and Chakraborty, T. (2023). Optimal rsu deployment using complex network analysis for traffic prediction in vanet. *Peer-to-Peer Networking and Applications*, 16(2):1135–1154.

Hota, L., Nayak, B. P., Kumar, A., Sahoo, B., and Ali, G. M. N. (2022). A performance analysis of vanets propagation models and routing protocols. *Sustainability*, 14(3):1379.

Ismail, N., Hossain, M. A., Noor, R. M., and Wahab, A. W. A. (2022). Enhanced congestion control model based on message prioritization and scheduling mechanism in vehicle-to-infrastructure (v2i). In *Journal of Physics: Conference Series*, volume 2312, page 012087. IOP Publishing.

Lim, K. L., Whitehead, J., Jia, D., and Zheng, Z. (2021). State of data platforms for connected vehicles and infrastructures. *Communications in transportation research*, 1:100013.

Lima, J. W. S. d., Callou, G., and Andrade, E. (2021). Queueing theory and stochastic petri net: A tutorial. *Research, Society and Development*, 10(3):e2810312826.

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

Maciel, P., Matos, R., Silva, B., Figueiredo, J., Oliveira, D., Fé, I., Maciel, R., and Dantas, J. (2017). Mercury: Performance and dependability evaluation of systems with

exponential, expolynomial, and general distributions. In *2017 IEEE 22nd Pacific Rim international symposium on dependable computing (PRDC)*, pages 50–57. IEEE.

Mahmood, D. A. and Horváth, G. (2020). Analysis of the message propagation speed in vanet with disconnected rsus. *Mathematics*, 8(5):782.

Mousa, R. J., Huszák, Á., and Salman, M. A. (2021). Enhancing vanet connectivity through a realistic model for rsu deployment on highway. In *Journal of Physics: Conference Series*, volume 1804, page 012104. IOP Publishing.

Nidhi and Lobiyal, D. (2021). Performance evaluation of rsus deployment at dense intersections. *International Journal of Information Technology*, 13(3):1095–1099.

RENASET (2024). Registro nacional de sinistros e estatísticas de trânsito em 2024.

Rodrigues, L., Gonçalves, I., Fé, I., Endo, P. T., and Silva, F. A. (2021a). Performance and availability evaluation of an smart hospital architecture. *Computing*, 103:2401–2435.

Rodrigues, L., Neto, F., Gonçalves, G., Soares, A., and Silva, F. A. (2021b). Performance evaluation of smart cooperative traffic lights in vanets. *International Journal of Computational Science and Engineering*, 24(3):276–289.

Shahin, R., Saif, S. M., El-Moursy, A. A., Abbas, H. M., and Nassar, S. M. (2023). Fog-rocl: A fog based rsu optimum configuration and localization in vanets. *Pervasive and Mobile Computing*, 94:101807.

Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2016). Edge computing: Vision and challenges. *IEEE internet of things journal*, 3(5):637–646.

Silva, L. G., Brito, C., Cardoso, I. B.-H. N., Sabino, A., Lima, L. N., Goncalves, G. D., Filho, G. P. R., Fé, I., and Silva, F. A. (2024). Desvendando a elasticidade de máquinas virtuais em vanets: Uma estratégia para aperfeiçoar o planejamento de capacidade em rsus. *Anais do XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2024)*.

Sinha, A. and Mishra, S. K. (2014). Queue limiting algorithm (qla) for protecting vanet from denial of service (dos) attack. *International Journal of Computer Applications*, 86(8):14–17.

Tan, H. and Chung, I. (2021). Rsu-aided remote v2v message dissemination employing secure group association for uav-assisted vanets. *Electronics*, 10(5):548.

Ullah, S., Abbas, G., Waqas, M., Abbas, Z. H., and Khan, A. U. (2023). Rsu assisted reliable relay selection for emergency message routing in intermittently connected vanets. *Wireless Networks*, 29(3):1311–1332.

Wang, Y., Song, Y., Cao, Y., Zhang, L., Ren, X., et al. (2023). Appeal-based distributed trust management model in vanets concerning untrustworthy rsus. In *2023 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE.

WIRE, B. (2018). U.s. autonomous car market 2018-2023 - there will be some 20.8 million autonomous vehicles in operation in the u.s. by 2030 - researchandmarkets.com.

Yu, H., Liu, R., Li, Z., Ren, Y., and Jiang, H. (2021). An rsu deployment strategy based on traffic demand in vehicular ad hoc networks (vanets). *IEEE Internet of Things Journal*, 9(9):6496–6505.