PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Programa de Pós-Graduação em Informática

Mariana Ramos de Brito

D-HOP:
protocolo dinâmico e distribuído para roteamento de veículos

Belo Horizonte

2016

Mariana Ramos de Brito

D-HOP:
protocolo dinâmico e distribuído para roteamento de veículos

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para obtenção do título de Mestre em Informática.

Orientador: Prof.ª Dr.ª Fátima de Lima Procópio Duarte-Figueiredo

Belo Horizonte

2016

FICHA CATALOGRÁFICA
Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

Brito, Mariana Ramos de
B862d D-HOP: protocolo dinâmico e distribuído para roteamento de veículos / Mariana Ramos de Brito. Belo Horizonte, 2016.
66 f. : il.

Orientadora: Fátima de Lima Procópio Duarte Figueiredo
Dissertação (Mestrado) - Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Informática.

1. Levantamentos de rotas. 2. Trânsito - Congestionamento. 3. Veículos a motor - Frotas. 4. Poluição. I. Figueiredo, Fátima de Lima Procópio Duarte. II. Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Informática. III. Título.

CDU: 681.3.03

Mariana Ramos de Brito

D-HOP:
protocolo dinâmico e distribuído para roteamento de veículos

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para obtenção do título de Mestre em Informática.

Prof.ª Dr.ª Fátima de Lima Procópio Duarte-Figueiredo – PUC Minas
(Orientadora)

Prof.ª Dr.ª Rossana Maria de Castro Andrade – Universidade Federal do Ceará
(Banca Examinadora)

Prof.ª Dr.ª Raquel Aparecida de Freitas Mini – PUC Minas (Banca Examinadora)

Prof.ª Dr.ª Anna Izabel João Tostes Ribeiro – PUC Minas (Banca Examinadora)

Belo Horizonte, 04 de maio de 2016.

AGRADECIMENTOS

Agradeço à Microsoft pelo apoio financeiro viabilizado por meio do projeto *Azure for Research*, e à Pontifícia Universidade Católica de Minas Gerais e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro.

Ao meu amigo Fabrício, aos outros amigos do mestrado e da graduação, e à minha orientadora, Fátima, pelas ideias e momentos de diversão.

À minha família, pelo apoio incondicional toda minha vida.

RESUMO

O congestionamento de veículos vem aumentando nas grandes cidades, causando perdas econômicas e poluição ambiental cada vez maiores. Redes veiculares surgiram para resolver ou, pelo menos, minimizar problemas de trânsito. Elas são formadas por veículos e infraestrutura fixa, podendo haver comunicação entre os veículos e as unidades que compõem a infraestrutura ou somente entre os veículos. Uma das aplicações possíveis em redes veiculares é a sugestão de rotas para os veículos, para aliviar congestionamentos existentes, distribuir o tráfego e diminuir o tempo de viagem e o gasto de combustível. Sugerindo rotas que consideram o trânsito de forma ampla, é possível evitar a formação de novos congestionamentos e diminuir os existentes, diminuindo também os efeitos negativos decorrentes desse problema tão comum dos centros urbanos. Nesse contexto, é proposto o D-Hop, um novo protocolo dinâmico e distribuído de roteamento de veículos. Ele leva em consideração o estado do tráfego das vias para calcular rotas menos congestionadas para os veículos. A infraestrutura fixa é responsável por avaliar o tráfego e guiar os veículos desde suas origens até seus destinos. Através do uso de controles específicos, realizados por cada módulo da rede veicular, o D-Hop propõe o envio de um número reduzido de mensagens entre veículos e infraestrutura. O D-Hop foi avaliado através de simulações, em diversos cenários. Os resultados mostraram que o D-Hop é melhor que o trabalho da literatura usado para comparação, ICODE, conseguindo diminuir o tempo de viagem dos veículos em 85%, em média. Com o controle no envio de mensagens proposto, até 95% menos mensagens são enviadas, em média, pela infraestrutura e até 67% menos pelos veículos, em média, em comparação a não usar o controle.

Palavras-chave: Redes Veiculares (VANETs). Sistema de transporte inteligente (ITS). Sugestão de rotas. Balanceamento de tráfego. Controle de congestionamento.

ABSTRACT

Vehicle congestion has increased in recent years, causing economic losses and environmental pollution. Vehicular networks have emerged to solve or at least minimize traffic problems. Vehicles and fixed infrastructure compose these networks, with communication happening between vehicles and infrastructure or only between vehicles. One possible application for vehicular networks is route recommendation, to relieve existing congestion, redistribute traffic and reduce travel time and fuel consumption. By suggesting routes that are aware of the traffic as a whole, it is possible to prevent congestion from happening and to reduce existing ones, and thus reduce negative effects of this problem, so common in urban centers. In this context it is proposed D-Hop, a new dynamic and distributed protocol for path recommendation. D-Hop considers the roads' traffic state to calculate less congested routes for vehicles. Fixed infrastructure is responsible for evaluating traffic and guiding vehicles, from their origin to their destination. Using specific controls, managed by each module, D-Hop reduces the number of messages sent between vehicles and infrastructure. D-Hop was evaluated through simulation in various scenarios. Obtained results showed D-Hop is better than the work from the literature used for comparison, reducing travel time of vehicles by up to 85%. Using the proposed message control up to 95% less messages were sent by infrastructure and up to 67% less by vehicles, compared to not using the control.

Keywords: Vehicular Networks (VANETs). Intelligent Transportation System (ITS). Path recommendation. Traffic balancing. Congestion control.

LISTA DE ABREVIATURAS E SIGLAS

DENATRAN – Departamento Nacional de Trânsito

GPS – *Global Position System*

MANETs – *Mobile Ad hoc Networks*

QoS – *Quality of Service*

RSU – *Road Side Unit*

RSUs – *Road Side Units*

TraCI – *Traffic Control Interface*

VANETs – *Vehicular Ad hoc Networks*

V2I – *Vehicle to Infrastructure*

V2V – *Vehicle to Vehicle*

V2X – *Vehicle to X*

XML – *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

O congestionamento no trânsito das grandes cidades é um problema, hoje em dia. As pessoas perdem muito tempo presas em congestionamentos, o que aumenta o nível de estresse e pode levar ao desenvolvimento de doenças. Além disso, uma grande quantidade de combustível é gasta, pois os carros continuam ligados enquanto parados, e a poluição ambiental sofre um aumento devido aos agentes poluentes liberados pelos veículos. Com o conhecimento sobre o fluxo de trânsito, é possível guiar os veículos para que eles evitem a área congestionada, sugerindo rotas alternativas (FAHMY; RANASINGHE, 2008). Dessa forma, congestionamentos existentes desaparecem de forma mais rápida e é possível evitar a formação de novos.

Nos últimos anos, tem havido um interesse crescente na área das redes veiculares, também conhecidas como *Vehicular Ad hoc Networks* (VANETs). Elas são formadas por veículos e infraestrutura fixa, geralmente localizada às margens das vias de trânsito, são caracterizadas por topologias altamente dinâmicas, enlaces intermitentes, e pelo fato de o movimento dos nós (veículos) ser restringido pelos limites das vias. Tais características levam a desafios como, por exemplo, a disseminação e o compartilhamento de dados, assim como questões de segurança (LIU; BI; YANG, 2009). Algumas das aplicações para as redes veiculares incluem: monitoração cooperativa do tráfego, auxílio a cruzamentos sem sinalização, prevenção de colisões, detecção de congestionamentos e sugestão de rotas. O acesso à Internet em qualquer lugar e a qualquer instante também é uma das utilizações esperadas (LI; WANG, 2007).

Com o uso de uma rede dedicada para resolver problemas de trânsito, como é o caso das VANETs, é possível coletar informações do trânsito em tempo real, independentemente de outras redes, tais como as de celular, Wi-Fi ou *Global Position System* (GPS), e usar essas informações para guiar os veículos evitando áreas de trânsito lento e impedir a formação de congestionamentos (WANG et al., 2015). Com o uso de um sistema criado exclusivamente para analisar o trânsito e tomar decisões que consideram o tráfego como um todo, não existe o problema de simplesmente mudar o congestionamento de lugar, o que diferencia métodos propostos para as VANETs de sistemas existentes, como o Google Maps, por exemplo. Além disso, com a distribuição do tráfego criada por tais métodos, é possível diminuir congestionamentos existentes e evitar que aconteçam outros, pois o sistema está constantemente vigiando o estado das vias de trânsito.

1.1 Motivação e justificativa

Atualmente, a quantidade de veículos em movimento nas ruas só tem aumentado, nas grandes cidades. Segundo dados do Departamento Nacional de Trânsito

(DENATRAN) (DENATRAN, 2016), a frota total do Brasil, em Janeiro de 2013, era de 76.588.058 veículos. Em Janeiro de 2014, a frota subiu para 82.060.911 veículos, um aumento de mais de cinco milhões de veículos. Com o aumento da frota, não apenas no Brasil, mas em todo o mundo, os congestionamentos tornaram-se um problema comum dos grandes centros urbanos. Os congestionamentos podem ocorrer a qualquer dia ou hora, causados por clima ruim, obras, semáforos defeituosos, eventos locais, acidentes, ou apenas pela pura quantidade de veículos em circulação que está em constante aumento, como é possível observar na Tabela 1.

Tabela 1 – Frota de veículos do Brasil nos meses de Janeiro

Ano	Veículos
2012	70.965.139
2013	76.588.058
2014	82.060.911
2015	87.073.671
2016	90.947.985

Fonte: Elaborada pela autora usando dados do DENATRAN (DENATRAN, 2016).

Há vários problemas consequentes dos congestionamentos, tais como tempo de viagem maior, aumento de custo por causa do gasto extra de combustível e produtividade perdida, visto que as pessoas, não só motoristas mas também as que utilizam o transporte público, perdem muito tempo no trânsito. Segundo a pesquisa "2015 *Urban Mobility Scorecard*" (INSTITUTE, 2015), em 2014, o custo do tempo e combustível extras para as áreas urbanas dos Estados Unidos foi de 160 bilhões de dólares, aproximadamente 580 bilhões de reais. Foram 6,9 bilhões de horas de tempo extra de viagem e 3,1 bilhões de galões a mais de combustível. Além disso, aproximadamente 41% dos atrasos acontecem fora dos horários de pico, ao contrário do que seria de se esperar.

Com a imprevisibilidade dos congestionamentos, fica clara a utilidade de um método de sugestão de rotas que tenha uma visão ampla do trânsito e distribua o fluxo de veículos assim que áreas lentas sejam detectadas. Dessa forma, essas áreas não se tornarão áreas de congestionamento. Usando a rede veicular, é possível fazer essa distribuição em tempo real, de forma dinâmica, com adaptações a cada mudança do tráfego.

1.2 Problema

Nesse contexto, o problema abordado neste trabalho é: como ter uma visão ampla do trânsito e guiar os veículos evitando áreas de trânsito lento para que todos consigam alcançar seus destinos da forma mais rápida possível? Para tanto, neste trabalho, é proposto um novo protocolo de sugestão de rotas. Ele utiliza a rede veicular e funciona de forma distribuída, avaliando o trânsito dinamicamente, para guiar os veículos evitando os congestionamentos existentes e a criação de novos. O D-Hop (***D**ynamic-**Hop***) utiliza

infraestrutura fixa, localizada nos cruzamentos, para manter tabelas de roteamento atualizadas com os tempos até os destinos, por cada caminho possível. Os tempos são calculados de acordo com as informações recebidas dos veículos, sendo, portanto, sempre atuais. Também foram propostos controles no envio de mensagens para diminuir o *overhead* da rede sem prejudicar o roteamento.

1.3 Objetivos

O objetivo principal deste trabalho é apresentar, testar e analisar o D-Hop, um método de sugestão de rotas que utiliza a rede veicular e funciona de forma distribuída e dinâmica, garantindo que os veículos sejam guiados desde sua origem até seu destino, da forma mais rápida possível. Especificamente, os objetivos foram propor controles para: diminuir o envio de mensagens para conseguir manter o roteamento atualizado de acordo com as informações atuais e também que os veículos ficassem sem rota o mínimo possível, sem sobrecarregar a rede. Também, diminuir o tempo de viagem e a consequente emissão de gás carbônico (CO_2) dos veículos, distribuir os veículos nas vias para garantir que não sejam formados congestionamentos.

1.4 Trabalho realizado e resultados obtidos

Para distribuir o tráfego pelas vias de trânsito, evitando congestionamentos, o roteamento dos veículos pode ser feito de forma centralizada ou distribuída, estática ou dinâmica (YOUNES; BOUKERCHE, 2014). Métodos centralizados apresentam problemas de gargalo e são pouco escaláveis, pois todos os veículos precisam se comunicar com um servidor central, que é responsável por juntar todas as informações, calcular rotas para os veículos e enviar essas rotas. Métodos estáticos conseguem calcular as melhores rotas para todos os veículos, entretanto, são incapazes de se adaptar imediatamente às mudanças do tráfego. O D-Hop foi criado para ser distribuído e dinâmico, pois, dessa forma, ele consegue adaptar o roteamento de acordo com o estado atual do trânsito.

Com o D-Hop, infraestruturas presentes em cada interseção vigiam o fluxo de veículos e atualizam tabelas de roteamento, que contêm os tempos até cada destino. Quando uma infraestrutura detecta mudanças significativas à sua volta, ela inicia o processo de atualização das tabelas, enviando mensagens para as outras infraestruturas sobre o novo tempo de viagem ao seu redor. Os veículos são responsáveis por pedir a rota pra infraestrutura, que recupera o caminho de menor tempo naquele instante para o destino solicitado. A resposta da infraestrutura é feita para todos os veículos no seu alcance de comunicação, para que menos mensagens de pedido precisem ser enviadas. Tanto o pedido de rota quanto as mensagens de atualização da tabela são controlados

para garantir o funcionamento do D-Hop com o menor número possível de mensagens.

O D-Hop foi implementado em uma rede veicular simulada e seus resultados foram comparados aos de uma adaptação do método da literatura ICODE, (YOUNES; ALONSO; BOUKERCHE, 2012), escolhido por ser diferente dos outros da literatura, e a outras versões de algoritmos, que são variações do D-Hop. As métricas avaliadas foram tempo de viagem, emissão de gás carbônico (CO_2), distância percorrida, veículos que não conseguiram receber resposta da infraestrutura antes de ficar sem caminho, número de mensagens de pedido de rota enviado pelos veículos, número de mensagens de atualização da tabela e velocidade média apresentada pelos veículos durante a simulação. O D-Hop apresentou resultados melhores que os outros métodos testados, com tempo de viagem 85% menor que a adaptação do método da literatura, em média. O número de mensagens de pedido de rota foi diminuído, em média, 67%, e da atualização da tabela em 95%, em média.

1.5 Organização do trabalho

O restante do trabalho está organizado em quatro capítulos. O Capítulo 2 contém conceitos necessários para a compreensão deste trabalho, assim como trabalhos relacionados. O método de roteamento de veículos proposto, D-Hop, é apresentado no Capítulo 3. Detalhes das simulações realizadas, os resultados obtidos e suas análises estão no Capítulo 4. Finalmente, no Capítulo 5, está a conclusão do trabalho, assim como sugestões para trabalhos futuros.

2 REFERENCIAL TEÓRICO E ESTADO DA ARTE

2.1 Redes veiculares

O interesse na área de redes veiculares nos últimos anos tem aumentado consideravelmente. Redes veiculares, também conhecidas como VANETs, são redes formadas por veículos e unidades fixas às margens das vias, havendo comunicação tanto entre os veículos como entre os veículos e a infraestrutura, através de comunicação sem fio. As redes veiculares caracterizam-se pela alta mobilidade dos nós (veículos), pelo fato de eles serem de vários tipos (carros, caminhões, ônibus etc.), pelo pouco tempo em que os nós ficam em contato, pela quantidade de nós que formam a rede e também pelo fato de os limites das vias restringirem o movimento dos nós. Essas características fazem com que os protocolos criados para outras redes sem fio não sejam adequados para as VANETs (ALVES et al., 2008).

As VANETs podem ser divididas em arquiteturas que expressam as organizações e formas de comunicação entre seus nós (veículos e unidades fixas) e têm várias aplicações possíveis, como, por exemplo, monitoração cooperativa do tráfego, auxílio a cruzamentos sem sinalização, detecção de congestionamentos, sugestão de rotas e prevenção de colisões. O acesso à Internet em qualquer lugar, e a qualquer instante, também é uma das utilizações esperadas para as redes veiculares (LI; WANG, 2007).

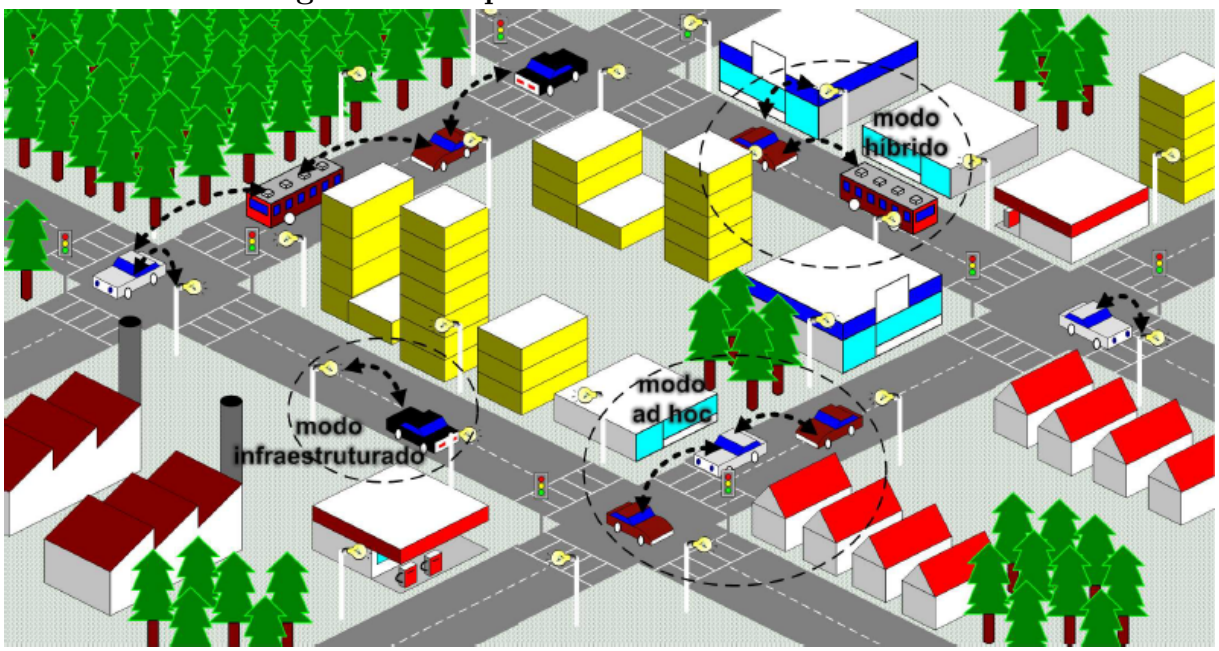
Vários motivos foram responsáveis pelo interesse crescente nessa área, como a extensa adoção do padrão IEEE 802.11 (rede sem fio); o fato de os fabricantes de carros estarem cada vez mais preocupados com a segurança, os aspectos ambientais e o conforto de seus veículos; e também o compromisso dos governos de vários países de alocar um espectro da rede *wireless* para a comunicação entre veículos (HARTENSTEIN; LABERTEAUX, 2008). A seguir serão detalhadas as arquiteturas das redes veiculares, assim como suas características, aplicações e desafios.

2.1.1 Arquiteturas

Redes veiculares são divididas em três arquiteturas (ALVES et al., 2009), de acordo com a forma como os nós se organizam e se comunicam. Essas arquiteturas são: (i) *ad hoc* puro, (ii) infraestruturada e (iii) híbrida, como é possível observar na Figura 1.

Na arquitetura *ad hoc* puro, também conhecida como *Vehicle to Vehicle* (V2V), os veículos comunicam-se apenas entre si, não existindo um nó centralizador ou elementos externos auxiliares. Nessa arquitetura, a conectividade da rede depende muito do fluxo de veículos, tendo em vista que cada um deles age como um roteador, enviando pacotes uns para os outros através de múltiplos saltos.

Figura 1 – Arquiteturas das redes veiculares



Fonte: Adaptada de (ALVES et al., 2009).

Na arquitetura infraestruturada, também conhecida como *Vehicle to Infrastructure* (V2I), são usados nós fixos às margens das vias (*Road Side Units* (RSUs)), servindo como pontos de acesso *wireless* e centralizando o tráfego da rede. Os veículos se comunicam apenas com os nós fixos e os nós fixos comunicam-se entre si. Em comparação com o modo V2V, a conectividade nessa arquitetura é maior e é possível acessar redes externas, como a Internet, por exemplo. Entretanto, para manter essa alta conectividade, é preciso um número grande de nós fixos, o que eleva o custo da rede. O D-Hop se encaixa nessa arquitetura.

Na arquitetura híbrida, também conhecida como *Vehicle to X* (V2X), que consiste na junção dos dois primeiros tipos, a comunicação entre os veículos é mantida, assim como uma quantidade mínima de nós fixos para aumentar a conectividade da rede. Dessa forma, é possível conseguir as vantagens das duas primeiras arquiteturas, mantendo a conectividade com um custo não tão alto.

2.1.2 Características

As redes veiculares têm características únicas que as diferenciam de outras redes móveis *ad hoc*. A rápida velocidade com que os nós se movem faz com que a topologia da rede seja altamente mutável, por isso os enlaces criados entre os nós da rede acontecem durante apenas alguns segundos, e se quebram rapidamente. Quando a densidade de veículos é baixa ou uma rota previamente estabelecida entre os nós para um pacote se quebra antes de uma nova rota ser formada, a probabilidade de a rede ficar desconectada aumenta. Isso faz com que os protocolos de roteamento usados em *Mobile Ad hoc Networks*

(MANETs) não sejam adequados para as VANETs (LIU; BI; YANG, 2009), o que cria a necessidade de novos protocolos serem criados apenas para as redes veiculares.

Outra característica interessante das redes veiculares é a possibilidade de modelar e prever a mobilidade dos seus nós, visto que o movimento deles é restrito pelos limites das vias e também pelas regras de trânsito. Isso torna possível prever quanto tempo vai durar um enlace (LIU; BI; YANG, 2009). Nas redes veiculares não existe uma grande preocupação com relação ao gasto de energia, visto que as baterias dos carros estão constantemente sendo recarregadas. Entretanto, atrasos na entrega dos pacotes são inviáveis dependendo da aplicação (LIU; BI; YANG, 2009). Se for um pacote de uma aplicação de segurança, por exemplo, o atraso na entrega de um único pacote pode fazer toda a diferença. Por isso a troca de pacotes deve ocorrer de forma rápida em uma rede veicular, usando protocolos especialmente criados para utilizar toda a capacidade da rede.

2.1.3 Aplicações

Existe uma lista extensa de potenciais aplicações para as redes veiculares. Essas aplicações podem ser divididas basicamente em duas classes: conforto e segurança (YOUSEFI; MOUSAVI; FATHY, 2006), embora os autores de (ALVES et al., 2009) defendam a separação das aplicações em três classes: segurança, entretenimento e assistência ao motorista. No caso da divisão em duas classes, a classe de assistência ao motorista é separada entre as outras duas.

As aplicações de segurança precisam divulgar informações rapidamente, para que o condutor do veículo tenha tempo de reagir, e geralmente têm sua divulgação limitada aos nós próximos à situação que causou o alerta. Algumas aplicações que se enquadram nessa categoria são: avisos de violação de sinais de trânsito, avisos sobre a velocidade do veículo em curvas, divulgação de informações sobre acidentes nas vias, divulgação de ocorrências no trânsito, auxílio para mudança de faixa, e avisos sobre condições adversas de ruas e estradas. As aplicações de segurança podem precisar tanto de comunicação V2V quanto de V2I, visto que elas têm diferentes características dependendo da aplicação (HARTENSTEIN; LABERTEAUX, 2008). Um ponto importante sobre as aplicações de segurança é que elas são sensíveis ao tempo, ou seja, precisam ser rápidas, e devem ter prioridade sobre outros tipos de aplicação (YOUSEFI; MOUSAVI; FATHY, 2006).

As aplicações de conforto são usadas para melhorar a experiência das pessoas nos veículos e a eficiência do tráfego. Também podem ser usadas para otimizar a rota até um destino. Exemplos dessas aplicações são: informações sobre o clima, informações turísticas, localização de postos de abastecimento, cobrança automatizada de pedágio, locais de estacionamento etc. Entretanto, a maioria delas volta-se para sistemas associados à Internet, como o compartilhamento de músicas e filmes, troca de mensagens instantâneas

e a distribuição de áudio e vídeo. Por terem essa natureza, as aplicações de conforto geralmente utilizam comunicação veículo-a-veículo (V2V), visto que seria muito caro manter uma infraestrutura toda conectada. Entretanto, o acesso à Internet, que é necessário para muitas dessas aplicações, só é possível através de pontos fixos, que são nós especiais da rede chamados *gateways* (ALVES et al., 2009).

É preciso verificar se as exigências de uma aplicação podem ser satisfeitas, e a que nível. Um fator importante nesse sentido é a taxa de utilização necessária para o funcionamento da aplicação, ou seja, quantos veículos precisam estar equipados com a tecnologia da rede veicular (HARTENSTEIN; LABERTEAUX, 2008) e também a quantidade de infraestrutura fixa necessária para que a aplicação funcione.

2.1.4 Desafios

Um dos desafios principais das redes veiculares é que um nó coordenador da comunicação não pode ser garantido (HARTENSTEIN; LABERTEAUX, 2008), visto que as VANETs são redes *ad hoc*. Embora algumas aplicações envolvam infraestrutura, é esperado que grande parte delas funcione de forma descentralizada. Isso faz com que o endereçamento na rede veicular se torne um problema, visto que, como os nós da rede são móveis, os endereços individuais de cada nó não são mantidos, tornando necessária uma forma de atribuição automática de endereços para que a comunicação entre os nós seja possível.

Outros desafios incluem a topologia altamente dinâmica das redes veiculares, devido ao movimento dos nós, e a propagação dos sinais de rádio, visto que a superfície metálica dos carros, os prédios e outras construções atrapalham o sinal. O fato de as redes veiculares precisarem se adaptar às diferentes condições do tráfego, seja uma grande quantidade de veículos ou pouca quantidade dos mesmos, também é um desafio, pois as aplicações precisam funcionar em todas as situações (LIU; BI; YANG, 2009).

A segurança na rede é outro desafio. É necessário que haja uma forma de garantir a veracidade da informação recebida, mas também é preciso garantir a privacidade de quem enviou a mensagem. A rede pode sofrer vários tipos de ataques, como simples escutas até modificações da mensagem original, prejudicando os usuários da rede (HARTENSTEIN; LABERTEAUX, 2008). Um agravante para falhas de segurança nas redes veiculares é que vidas podem ser postas em risco.

Existe também o desafio da implantação da rede. É necessário que as pessoas equipem seus veículos com a tecnologia para a rede, mas o custo só diminuiria caso mais pessoas fizessem o mesmo. Também é necessário certo número de veículos equipados para que a rede tenha condições de funcionar, portanto, não existem muitos incentivos para

as primeiras pessoas que equiparem seus veículos. É provável que nós fixos às margens das vias sejam usados para atrair os primeiros usuários (HARTENSTEIN; LABERTEAUX, 2008), embora nem todas as aplicações de uma rede veicular possam funcionar sem a comunicação direta entre os nós.

Modelar o movimento dos veículos em uma rua/estrada também é uma tarefa difícil (FIORE et al., 2007), que depende de diversos fatores. O formato da via, a densidade do tráfego, o comportamento dos motoristas etc. Tudo isso contribui para que modelar o tráfego não seja trivial. Outro desafio é sobre o uso eficiente do canal de comunicação, já que é difícil conseguir banda *wireless* disponível (LIU; BI; YANG, 2009) e as aplicações das VANETs usam muitos *broadcasts* e *multicasts* para comunicação.

2.2 Simulação

Para realizar testes com veículos reais pode ser necessário um grande número de pessoas, muitos veículos equipados, criar e instalar infraestrutura fixa, os testes podem ser perigosos (quando testando aplicações de segurança), precisar de condições climáticas específicas e ambientes favoráveis. Fora isso, para garantir os resultados pode ser necessário repetir experimentos, o que aumenta ainda mais o custo de testes reais das aplicações de redes veiculares.

Por essa razão, simulação é a forma escolhida para testar uma rede veicular e suas aplicações atualmente. Entretanto, reproduzir as condições reais do tráfego em uma simulação é uma tarefa complexa, que envolve modelar a propagação de sinais, a disputa pelo acesso ao meio, entre outros protocolos de redes (ALVES et al., 2009). E também é necessário criar modelos de mobilidade específicos, que representem o movimento dos veículos.

Para simular a comunicação em um ambiente veicular, tanto a mobilidade dos veículos quanto as comunicações entre eles e a infraestrutura precisam ser modeladas usando as plataformas adequadas (SHAFIEE et al., 2011). Simuladores distintos são utilizados para modelar a rede e a mobilidade.

A saída do simulador de mobilidade precisa ser convertida para um formato que o simulador de rede possa ler e para isso existem várias ferramentas de conversão. Alguns tipos de aplicação, como aplicações de segurança, precisam que os simuladores de rede e mobilidade estejam em constante interação, pois o movimento dos veículos é definido por essas aplicações. Para esse último tipo de aplicações, uma plataforma de integração entre os simuladores é necessária.

2.2.1 Simulador de mobilidade

Um simulador de mobilidade bastante usado para a simulação de redes veiculares é o SUMO (SUMO, 2016). Ele é um simulador microscópico, ou seja, modela comportamentos individuais dos veículos, *open-source*, desenvolvido para lidar com grandes redes de estradas, e suporta tipos diferentes de veículos, mudança de pista em ruas com várias pistas, rotas dinâmicas de veículos, entre outras características. Outros simuladores de mobilidade incluem: CityMob (MARTINEZ et al., 2008), STRAW (STRAW, 2008), e VanetMobiSim (VANETMOBISIM, 2006), todos grátis e *open-source*.

Para simular o tráfego de veículos, o simulador de mobilidade precisa da rede das ruas e estradas, que é representada por um conjunto de nós (interseções) ligados uns aos outros por arestas (estradas/ruas). Diversas informações sobre os nós e as arestas da rede são necessárias, por exemplo, a localização dos nós, o número de faixas das arestas, os sentidos permitidos nas arestas etc. Outras informações, como sinais de trânsito e velocidades máximas permitidas, também podem ser adicionadas. Ainda, é necessária a demanda do tráfego, que vai gerar os fluxos de veículos na rede. Os veículos movem-se das suas origens aos seus destinos e os comportamentos individuais de cada veículo podem ser especificados, seguindo um modelo de mobilidade.

Existem várias maneiras de construir a rede para o SUMO, como a partir de um arquivo *Extensible Markup Language* (XML) escrito manualmente descrevendo a rede, usando o gerador automático disponível com o SUMO, ou convertendo mapas reais para o formato lido pelo SUMO. Depois de definir a rede, a demanda do tráfego precisa ser gerada. Isso pode ser feito de várias formas no SUMO, entre elas: definir viagens (origem e destino de cada veículo, assim como o instante em que ele começou a viagem), fluxos (vários veículos seguindo a mesma viagem) ou rotas (deterministas, randômicas ou importadas de outras ferramentas de simulação).

2.2.2 Simulador de rede

Simuladores de rede são responsáveis por simular detalhadamente origem, destino, transmissão, recepção, rotas, *links* e canais, a nível de pacote. Alguns exemplos incluem: GloMoSim (ZENG; BAGRODIA; GERLA, 1998), GTNetS (GTNETS, 2008), JiST/SWANS (JIST/SWANS, 2004), ns-2 (NS-2, 2012), ns-3 (NS-3, 2014) e SNS (SNS, 2001). Entretanto, a maioria dos simuladores de rede é desenvolvida para simular MANETs, portanto eles precisam de extensões para que seja possível simular VANETs (MARTINEZ et al., 2011).

O OMNeT++ (OMNET++, 2016) é um ambiente de simulação de eventos discretos, bastante usado para simular redes veiculares em conjunto com o SUMO. Seus modelos são formados por componentes, que são responsáveis por simular as diversas partes das

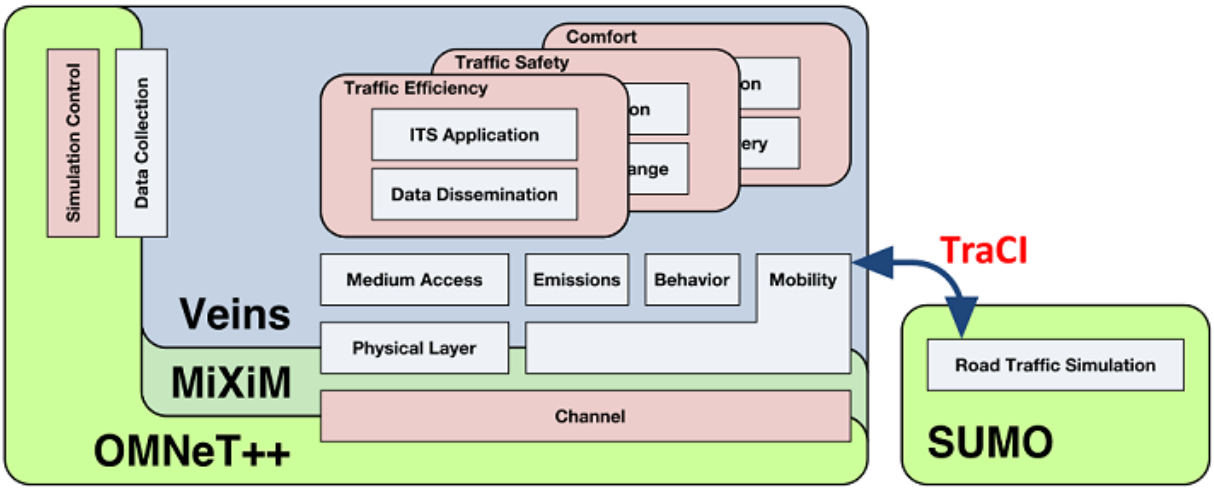
redes, reutilizáveis programados em C++ e é possível juntá-los em componentes e modelos maiores utilizando a linguagem de alto nível *NED*, própria do OMNeT++.

Para implementar o cenário de simulação, os algoritmos dos componentes são executados pelo modelo. As mensagens trocadas pelos componentes representam quadros ou pacotes e podem ser enviadas diretamente para seus destinos ou para outros componentes, seguindo um caminho pré-definido. Os componentes e a topologia da rede são definidos em arquivos *NED*, enquanto seus comportamentos são adicionados por arquivos em C++ e seus parâmetros são lidos de arquivos de inicialização (*ini*). O OMNeT++ também oferece um ambiente gráfico para as simulações, com animação para os componentes da rede e informações detalhadas dos pacotes sendo enviados. É possível aumentar a funcionalidade do OMNeT++ usando *frameworks* externos, que podem ser importados para o OMNeT++. É este o caso para a simulação de uma rede veicular.

2.2.3 Integração dos simuladores

Para que os veículos controlados pelo simulador de mobilidade possam mudar de caminho dependendo das decisões tomadas com o uso da rede gerada pelo simulador de rede, é preciso que esses dois simuladores estejam funcionando ao mesmo tempo e suas informações sejam trocadas em tempo real. A plataforma de integração que cuida da interação entre o SUMO e o OMNeT++ é o Veins (VEINS, 2016), um *framework* de simulação de comunicação interveicular *open-source* que é importado para dentro do OMNeT++.

Figura 2 – Arquitetura do Veins



Fonte: Adaptada de (VEINS, 2016).

A comunicação em tempo real, necessária quando o comportamento dos veículos precisa mudar durante a execução devido a um acontecimento na rede, é feita através do protocolo *Traffic Control Interface* (TraCI), parte do Veins, que faz com que cada veículo simulado pelo SUMO vire um nó na rede simulada pelo OMNeT++ e permite

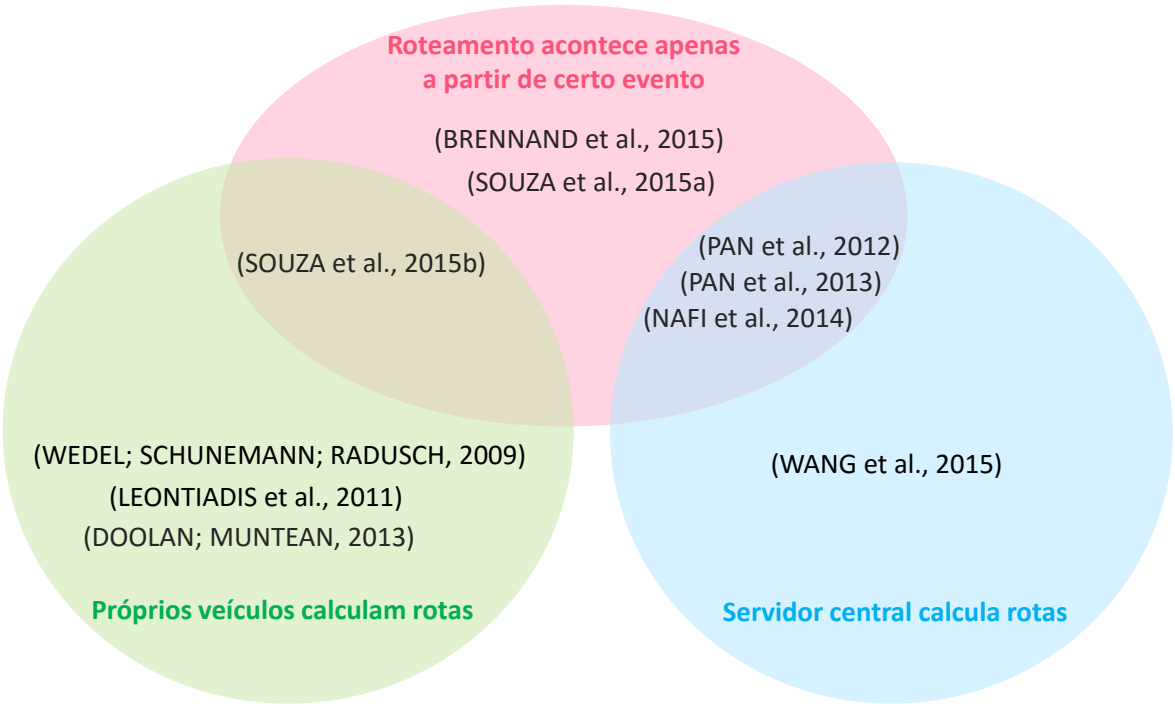
ao algoritmo funcionando no OMNeT++ dar ordens aos veículos simulados pelo SUMO. Na Figura 2 está uma ilustração de como funciona a integração entre o OMNeT++ e o SUMO, através do TraCI.

O Veins utiliza o MiXiM, que é o *framework* para o OMNeT++ que oferece modelos de redes *wireless* móveis e fixas, para oferecer modelos detalhados do padrão IEEE 802.11p (padrão *wireless* das redes veiculares), incluindo acesso ao meio com qualidade de serviço (*Quality of Service* (QoS)), efeitos de ruído, efeitos de interferência na rede e outros. Várias informações da simulação são disponibilizadas pelo Veins como, por exemplo, tempo gasto em uma viagem, emissões de gases do efeito estufa feitas pelos veículos, distância total percorrida pelo veículo, velocidades máxima e mínima alcançadas e outras. O Veins suporta em seus cenários: construções, limites de velocidade, quantidades diferentes de pistas, sinais de trânsito, restrições de acesso às vias e outros.

2.3 Trabalhos relacionados

Foram notadas certas similaridades entre os trabalhos da literatura relacionados ao D-Hop. A partir dessas similaridades, foi criada a Figura 3, que ilustra a separação dos trabalhos relacionados de acordo com sua característica principal.

Figura 3 – Similaridades dos trabalhos relacionados



Fonte: Elaborada pela autora.

As características notadas foram: (i) existência de um servidor central responsável por calcular rotas para todos os veículos, o que pode levar a problemas de gargalo; (ii) os próprios veículos são responsáveis por calcular novas rotas para si mesmos, o que pode

fazer o congestionamento simplesmente mudar de lugar; e (iii) o roteamento acontece apenas a partir de certo evento, como a detecção de um congestionamento ou acidente, o que atrasa a resolução do congestionamento.

2.3.1 Roteamento acontece apenas a partir de certo evento

Com RSUs responsáveis por manter grafos onde as arestas são as ruas, os nós são interseções e o peso das arestas é inversamente proporcional à velocidade atual da rua correspondente, em (BRENNAND et al., 2015), os autores propuseram um sistema de controle de tráfego que funciona de forma periódica. Os veículos enviam informações do tráfego para as RSUs, que mantêm seus grafos atualizados. Em intervalos fixos, as RSUs verificam se existe um congestionamento através do seu grafo e, caso sim, calculam os k menores caminhos para os quais desviar os veículos para que eles não passem pelo congestionamento. Para evitar que todos os veículos sejam roteados para o mesmo lugar, é usada uma distribuição de probabilidade para escolher qual dos k caminhos será o escolhido no momento. Uma redução de 23% foi observada no tempo de viagem dos veículos seguindo o método proposto, mas a detecção, e consequente roteamento dos veículos para evitá-lo, não é feita em tempo real, o que permite que os congestionamentos existentes cresçam até o próximo período de detecção.

Em (SOUZA et al., 2015a), é proposto um sistema de roteamento de veículos que precisa de, pelo menos, uma RSU para funcionar. As RSUs mantém grafos da sua área de cobertura, sendo as ruas, arestas, e as interseções, nós. Cada aresta tem um peso, que é inversamente proporcional à velocidade média calculada para a rua correspondente. A velocidade média é calculada a partir das mensagens recebidas dos veículos. A partir desse grafo, e sabendo quais arestas apresentam congestionamento, as RSUs controlam o fluxo de veículos impedindo que veículos passem por tais arestas. Os autores compararam seu sistema com os métodos propostos em (BRENNAND et al., 2015) e em (PAN et al., 2012), obtendo tempos de viagem menores. Entretanto, o cálculo das novas rotas é feito para cada veículo individualmente seguindo um algoritmo guloso, que escolhe a aresta de menor peso em cada interseção, até o destino do veículo. Esse cálculo pode se tornar muito custoso com o aumento do número de veículos, fazendo com que muitos não consigam receber a rota antes de chegar ao local do congestionamento.

2.3.2 Próprios veículos calculam rotas

Em (WEDEL; SCHUNEMANN; RADUSCH, 2009), os autores propõem um algoritmo que calcula rotas evitando vias congestionadas. Os veículos executando o algoritmo proposto transmitem suas velocidades médias ao passar por cada segmento de via para os veículos vizinhos. Com essas informações recebidas dos vizinhos, os veículos

calculam o peso dos segmentos de via. Os pesos dos segmentos são usados pelos veículos para recalcular suas rotas, escolhendo a rota com o menor tempo de viagem. Os resultados obtidos pelos autores, através de simulação, demonstram que os tempos de viagem foram reduzidos em quase 50% com 80% dos veículos utilizando o algoritmo proposto. Entretanto, os veículos só conseguem pegar informações sobre os segmentos que estão perto deles, o que pode fazer com que o congestionamento simplesmente mude de lugar. Este é o mesmo problema encontrado no método EcoTrec, proposto em (DOOLAN; MUNTEAN, 2013). Usando o EcoTrec, os veículos calculam para si próprios rotas alternativas, que gastem a menor quantidade de combustível, usando o algoritmo de Dijkstra com as informações recebidas dos veículos vizinhos. Portanto, muitos veículos podem se dirigir ao mesmo caminho, criando congestionamentos.

Em (LEONTIADIS et al., 2011), os próprios veículos são responsáveis por coletar informações do trânsito e compartilhar entre si. Cada um estima o tráfego dos segmentos e decide, por conta própria, para onde ir, usando uma modificação do algoritmo de Dijkstra com o peso das arestas sendo a estimativa de tráfego calculada. Embora os resultados obtidos mostrem que o método proposto consegue reduzir congestionamentos existentes, pode acontecer de eles simplesmente mudarem de lugar, pois os veículos não conseguem ter uma visão ampla do trânsito sendo responsáveis pelo processo de roteamento inteiro.

Os autores de (SOUZA et al., 2015b) propuseram um sistema de roteamento que funciona sem a necessidade de infraestrutura fixa, o GARUDA. Um veículo que sofre um acidente começa a enviar mensagens para os outros, que decidem até quando propagar a mensagem do acidente e se devem mudar de rota. Caso a rota dos veículos que receberam essa mensagem fosse passar pelo local do acidente, esses veículos calculam todas as rotas alternativas e escolhem a menor rota, usando Dijkstra. O sistema proposto foi comparado com os métodos propostos em (PAN et al., 2012) e apresentou tempos de viagem menores. No entanto, não apenas o GARUDA só entra em ação quando acontece um acidente, como também deixa a decisão do roteamento ser tomada pelos veículos. Congestionamentos causados por outras razões não podem ser resolvidos pelo GARUDA e o congestionamento criado pelo acidente pode simplesmente mudar de lugar, dessa vez causado pelo volume de veículos que escolhem a mesma rota alternativa.

2.3.3 Servidor central calcula rotas

Usando redes veiculares em conjunto com a rede de celular, os autores de (WANG et al., 2015) propõem um algoritmo que calcula o melhor caminho para certo destino em tempo real, levando em consideração as preferências dos motoristas. RSUs instaladas ao longo das vias de trânsito coletam informações dos veículos e enviam essas informações para um servidor, que é responsável por calcular os caminhos considerando o trânsito

de forma global, através do algoritmo proposto, que é baseado em uma otimização de Lyapunov. Os resultados, obtidos através de simulação, mostram que utilizando o algoritmo proposto o tempo de viagem é reduzido em comparação a não usar os caminhos planejados. Entretanto, o uso de um servidor central que calcula as rotas pode ser problemático, pois tudo depende dele. Podem acontecer problemas de gargalo ou o servidor cair.

Os autores de (PAN et al., 2012) propõem três estratégias de roteamento de veículos: DSP (*Dynamic Shortest Path*), que calcula o menor caminho atualmente até o destino, RkSP (*Random k Shortest Paths*), que calcula os k menores caminhos e escolhe um randomicamente, e EBkSP (*Entropy Balanced k Shortest Paths*), no qual são calculados k menores caminhos e escolhido o de menor popularidade definida pela entropia dos caminhos. O sistema proposto funciona com base em um serviço central de monitoramento que pode ser dividido entre vários servidores, que é responsável por analisar as informações do trânsito coletadas pelos veículos e por infraestrutura fixa, se existir, e calcular novas rotas para veículos específicos caso detecte situações de congestionamento. Os resultados, obtidos através de simulação, mostram que as três estratégias são melhores que não usar roteamento algum e que o EBkSP é o melhor de todos, conseguindo reduzir mais o tempo de viagem. No sistema proposto, no entanto, as estratégias de roteamento só são usadas quando um congestionamento é detectado, fora que as decisões e cálculos para o roteamento dependem do serviço central. Se o roteamento existir o tempo inteiro, a tendência é que congestionamentos sejam pequenos ou nem aconteçam.

Continuando o trabalho de 2012, em (PAN et al., 2013) os autores propõem mais duas estratégias: uma modificação do algoritmo A* de menores caminhos, AR* (*A* Shortest Path with Repulsion*), que considera tanto o tempo de viagem quanto os caminhos dos outros veículos no cálculo do menor caminho, e FBkSP (*Flow Balanced k Shortest Paths*), que calcula k menores caminhos para cada veículo e escolhe o caminho que minimiza o impacto no trânsito naquela região da rede. Os resultados foram obtidos por simulação e mostram que o AR* obtém os menores tempos de viagem, seguido pelo EBkSP e pelo FBkSP, apresentando, entretanto, um custo maior para ser computado. Como no trabalho anterior, o sistema proposto depende de vários parâmetros, o que cria a necessidade de ajustá-los bem para garantir que o sistema funcione. Os autores pretendem dar continuidade ao trabalho, para que esses parâmetros possam se auto ajustar no sistema.

Usando um algoritmo de predição linear modificado, os autores de (NAFI et al., 2014) propõem um sistema de gerenciamento de tráfego que prevê qual será a intensidade do tráfego em diferentes interseções. A partir da predição, um controlador central roteia os veículos e muda os ciclos dos semáforos para reduzir congestionamentos. RSUs instaladas nas interseções coletam as informações do trânsito recebidas dos veículos e passam esses dados para o controlador central. Usando simulação, os autores compararam o fluxo

previsto com o fluxo real. O fluxo previsto ficou parecido com o real, mas, quando aconteceram picos no fluxo real, a previsão não conseguiu se adaptar e ficou mais distante da realidade. Os autores também analisaram o tempo de viagem e o tempo de espera nas interseções com e sem o método proposto. Usando o método proposto, ambos foram reduzidos. Parecido com (PAN et al., 2012), a decisão de iniciar o roteamento depende de ter sido previsto um congestionamento, em vez de deixar o roteamento existir desde sempre, o que cria a necessidade de se encontrar um bom limite entre o que pode ser considerado congestionamento e o que não pode. Também há uma grande troca de pacotes entre os módulos da rede, pois as RSUs enviam pacotes para os veículos a cada um segundo e para o controlador central a cada um minuto, e os veículos enviam respostas para a RSU a cada cinco segundos.

2.3.4 Principais trabalhos relacionados

Para melhor distribuir o tráfego de veículos pelo mapa em (YOUNES; BOUKERCHE, 2014), os autores evitam que os segmentos de saída dos cruzamentos sejam sobrecarregados por veículos que estejam se dirigindo para uma mesma área. Os veículos são conduzidos salto-a-salto pelos cruzamentos, sempre seguindo o caminho que gaste menor tempo total até o destino. Caso seja calculado que o segmento de menor tempo no momento está saturado de veículos, o próximo segmento com menor tempo será o escolhido. Dessa forma, em cada cruzamento o tráfego é distribuído para evitar que segmentos fiquem congestionados. Os autores compararam os resultados do novo método, Bal-Traf, com os resultados do trabalho anterior, (YOUNES; ALONSO; BOUKERCHE, 2012), descrito a seguir. Embora o Bal-Traf tenha apresentado um aumento no tempo de viagem e na distância percorrida, ele conseguiu eliminar totalmente os segmentos sobrecarregados e diminuiu em 5% a densidade em cada segmento. Entretanto, visto que tanto o tempo de viagem quanto a distância percorrida aumentaram, não houve uma melhora notável no trânsito apenas por não sobrecarregar segmentos.

Em (YOUNES; ALONSO; BOUKERCHE, 2012; YOUNES; BOUKERCHE, 2013; YOUNES et al., 2015), os autores propõem um novo protocolo dinâmico em tempo real para encontrar o caminho mais rápido até certo destino, de maneira distribuída, o ICODE (*Infrastructure-based Congestion Avoidance Protocol*). Considerando a existência de unidades fixas (RSUs) em cada esquina, elas são responsáveis por informar um caminho alternativo para cada destino dos veículos na sua vizinhança. As RSUs conversam entre si para garantir sempre o menor tempo de viagem possível para cada destino e guardam tabelas de roteamento com a próxima *Road Side Unit* (RSU) para qual os veículos devem se dirigir para cada destino desejado. O roteamento é feito salto-a-salto, com os veículos indo de RSU em RSU. Para comparar com o ICODE, os autores propuseram dois outros métodos, *Simple*, no qual a tabela é preenchida uma única vez e não muda, e *Shortest*,

no qual um caminho será atualizado caso o novo recebido seja menor que o armazenado. Os resultados, obtidos através de simulação, mostram que o protocolo proposto é sempre o que sugere os caminhos de menor tempo de viagem até os destinos, embora envie mais mensagens que os outros dois. Os autores, no entanto, testaram o método proposto com apenas três destinos possíveis, o que não é muito realista.

O ICODE, apesar de ser um método interessante, diferente dos outros da literatura, e diminuir o tempo de viagem, apresenta alguns problemas. Dois deles estão relacionados ao envio de mensagens por parte das RSUs: (i) as RSUs são responsáveis por fazer propaganda de todos os destinos periodicamente, o que causa um *overhead* enorme na rede quando existem mais destinos possíveis que apenas os três testados pelos autores; e (ii) os destinos possíveis também enviam mensagens periódicas para a atualização da tabela, ou seja, mesmo que a situação do trânsito perto de um destino esteja estável, o destino continua causando a atualização das tabelas de todas as RSUs periodicamente, sem necessidade. Outro problema notado no ICODE é sobre as informações armazenadas na tabela de roteamento e seu consequente envio para os veículos: apenas o próximo salto (um salto) é guardado e enviado, portanto, os veículos sempre precisam receber uma nova mensagem, caso contrário ficam sem rota.

Para corrigir os problemas apresentados pelo ICODE, e apresentar outras melhorias, neste trabalho foi proposto o D-Hop. O D-Hop é um protocolo de roteamento de veículos dinâmico e distribuído, que se adapta às mudanças do tráfego, e tem controle no envio de todas as mensagens. Cinco saltos de cada vez são guardados e enviados para os veículos, e é realizado um balanceamento dos veículos pelos caminhos de saída, para evitar criar congestionamentos. Por ser distribuído, não tem possibilidade de o D-Hop ter problemas de gargalo ou de queda de um serviço central que controle o roteamento. Sendo dinâmico, o D-Hop consegue perceber qualquer alteração no tráfego rapidamente e adaptar o roteamento para evitar a formação de um congestionamento ou diminuir algum congestionamento que tenha começado.

3 D-HOP

Neste capítulo é apresentado o método proposto neste trabalho, o D-Hop. Ele consiste em um protocolo de roteamento de veículos que funciona de forma dinâmica, redirecionando o tráfego, salto-a-salto, para garantir que todos os veículos consigam alcançar seu destino da forma mais rápida possível. O D-Hop segue a ideia do ICODE, descrito anteriormente, e depende da presença de unidades fixas (RSUs), nas interseções, para seu funcionamento. As RSUs controlam o tráfego através de tabelas de roteamento com os destinos possíveis, armazenando o tempo estimado de viagem, a partir delas, até cada destino.

Os veículos são direcionados para o caminho que tenha o menor tempo estimado no momento. Cada RSU vigia o fluxo de veículos à sua volta, estimando o tempo que um veículo leva para percorrer aquela região. Se uma alteração considerável no tempo é detectada, a RSU inicia o processo de atualização da tabela, avisando às outras RSUs sobre a mudança no tempo de viagem. Dessa forma, todas as RSUs sempre sabem os menores tempos de viagem até cada destino. Cada RSU atualiza as RSUs vizinhas e essas, por sua vez, suas vizinhas. Assim, todas as RSUs de uma determinada região recebem a informação sobre alteração no tempo de viagem do segmento que gerou a mensagem original.

O processo de roteamento começa com o veículo, que envia pedidos de rota para a RSU responsável pelo segmento no qual ele se encontra. O pedido de rota é controlado por cada veículo, para evitar que a rede seja sobrecarregada de mensagens. O veículo utiliza a sua velocidade e o número de saltos que ele ainda tem armazenado para impedir o envio de mensagens desnecessárias. Assim que uma RSU recebe um pedido de rota, ela vai procurar o caminho de menor tempo até o destino desejado e enviar para os veículos, que vão, então, se dirigir ao caminho informado.

A seguir, serão detalhadas as etapas do roteamento de veículos realizado pelo D-Hop. Inicialmente, há o processo ilustrado de roteamento de um veículo, desde sua origem até seu destino, seguindo o D-Hop. A seguir, é explicado o conceito de segmento de via e como ele se relaciona com as RSUs. Então, a tabela de roteamento e a atualização feita pelas RSUs serão discutidas. Finalmente, é detalhado o processo de roteamento, que envolve o controle do pedido de rota dos veículos e a resposta da RSU.

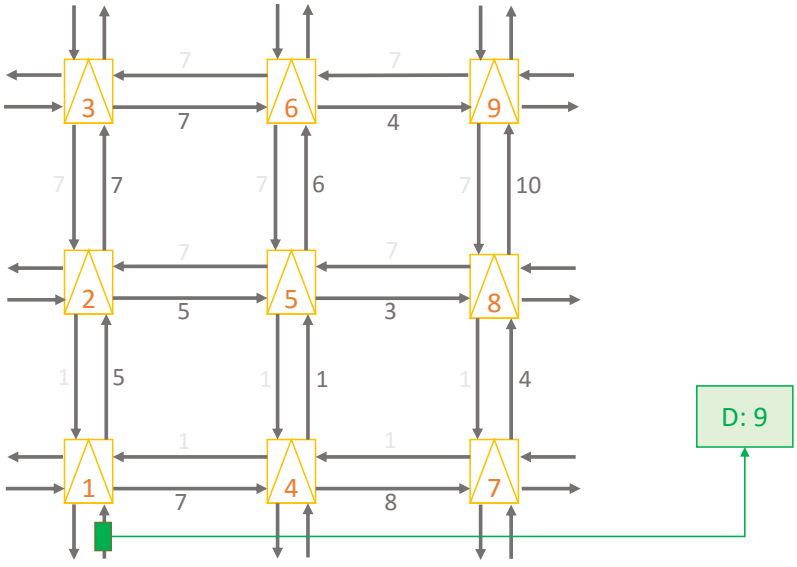
3.1 Processo de roteamento seguindo o D-Hop, ilustrado

Os segmentos e seus sentidos são representadas pelas setas. Os números perto de cada seta indicam o tempo estimado para percorrer aquele segmento, sendo que os

que não são importantes para o exemplo estão em cor mais clara para não atrapalhar a visualização. Nos quadrinhos, D é o destino, T é o tempo estimado até o destino ou para percorrer um segmento, C é o caminho (saltos) até o destino, S é o segmento sobre o qual é a mensagem. O veículo para exemplo é o retângulo verde-escuro. Ele envia mensagens de *WantHop* (pedido de rota), em verde, para as RSUs no seu caminho. Os saltos que o veículo ainda sabe aparecem nos quadrinhos roxos.

As RSUs são os triângulos dentro de retângulos e podem aparecer em três cores: (i) laranja: estado normal; (ii) vermelho: buscando na tabela de roteamento o menor caminho e respondendo a *WantHop* recebida de um veículo (os segmentos do caminho recuperado também aparecem em vermelho nesse momento); e (iii) azul: enviando ou retransmitindo mensagem de propaganda sobre segmentos cujo tempo estimado de viagem mudou (o segmento que mudou de tempo também aparece em azul nesse momento). As mensagens enviadas pelas RSUs também aparecem em três cores: (i) vermelho: *NextHop* (resposta de rota), enviada para o segmento de onde veio uma *WantHop*; (ii) azul: mensagem de propaganda inicial, criada pela RSU que cuida do segmento que mudou de tempo, avisando às outras sobre a mudança; e (iii) laranja: mensagem de propaganda retransmitida, enviada por cada RSU às vizinhas, somando o novo tempo até o destino por cada um de seus segmentos (que aparecem em laranja).

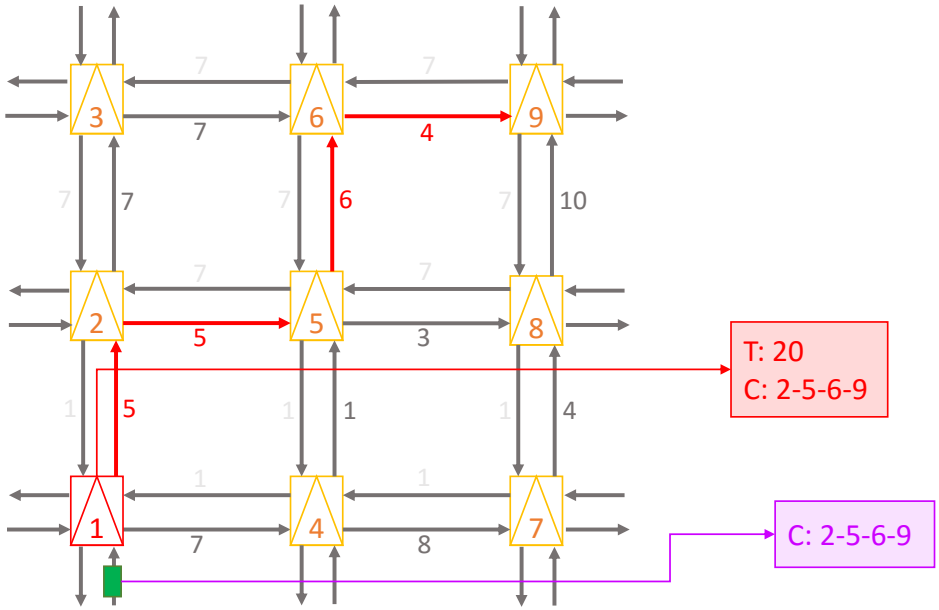
Figura 4 – Veículo envia *WantHop* à RSU 1, pedindo caminho até a 9



Fonte: Elaborada pela autora.

Na Figura 4, o veículo pede à RSU 1, que controla o segmento que ele está, o caminho até seu destino, a RSU 9. A RSU 1 recupera na sua tabela de roteamento o caminho de menor tempo até o destino e envia por *broadcast* para o segmento de onde veio o *WantHop*. Na Figura 5, está marcado em vermelho o caminho de menor tempo no momento até o destino 9, começando na RSU 1. O veículo recebe a mensagem e guarda o caminho.

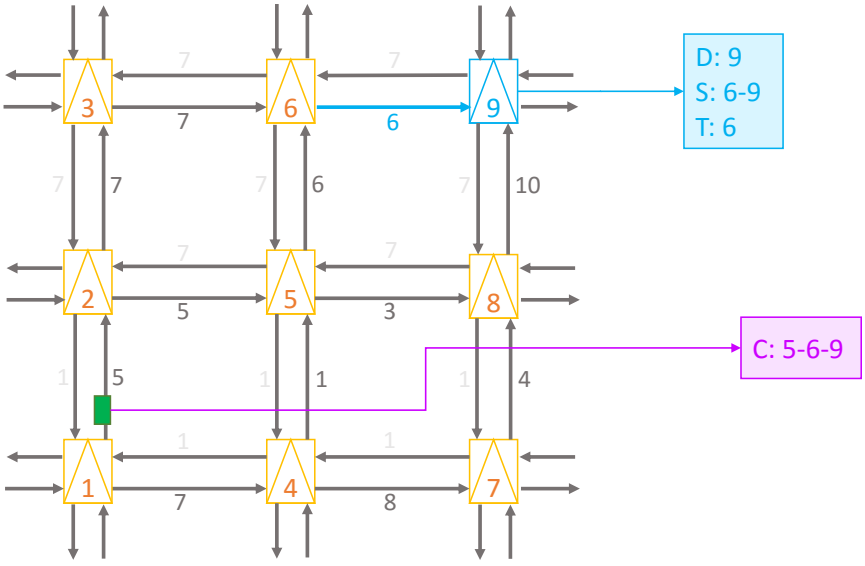
Figura 5 – RSU 1 recupera caminho de menor tempo e informa ao veículo



Fonte: Elaborada pela autora.

Na Figura 6, a RSU 9 percebe que o tempo estimado para percorrer o segmento 6-9 mudou. Portanto, ela envia uma mensagem de propaganda sobre esse segmento, indicando ela própria como destino e informando o novo tempo às suas vizinhas. O veículo continua seguindo seu caminho armazenado.

Figura 6 – RSU 9 percebe mudança no tempo do segmento 6-9

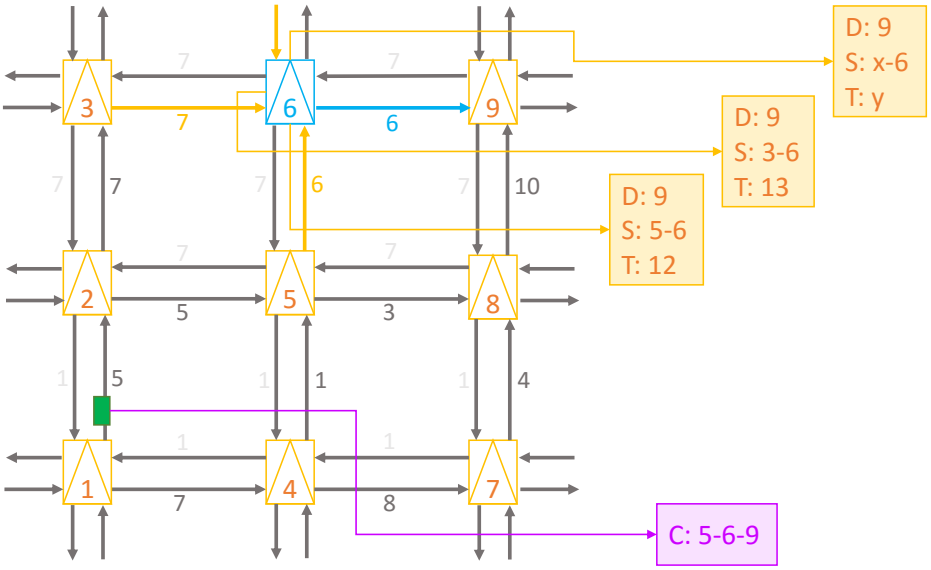


Fonte: Elaborada pela autora.

Na Figura 7, estão exemplificadas três das mensagens de retransmissão que a RSU 6 envia sobre o segmento da RSU 9 que mudou de tempo. Uma mensagem de retransmissão é enviada para cada segmento das RSUs que retransmitem a mensagem. A mensagem sobre o segmento 9-6 não apareceu na figura para que ela não ficasse muito confusa. O

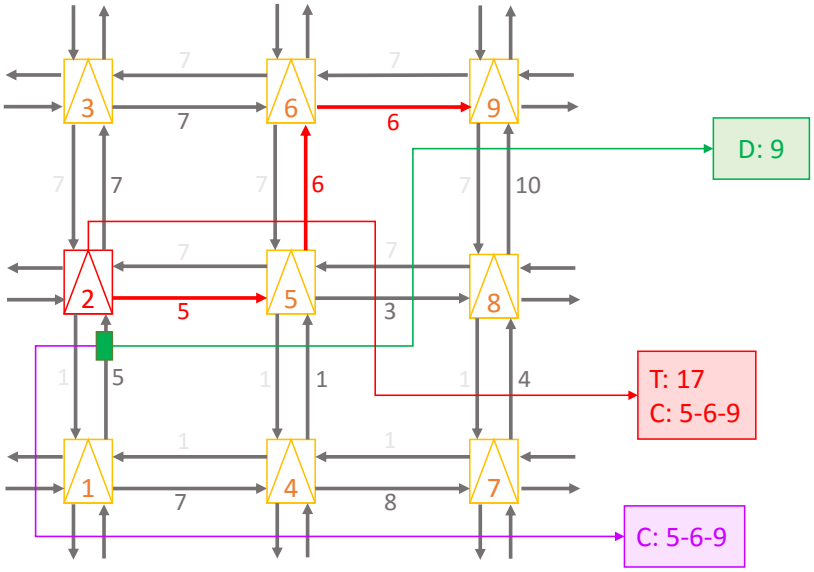
caminho até cada destino é montado dessa forma, de salto em salto, e os tempos estimados de cada segmento vão sendo somados como tempo até o destino.

Figura 7 – RSU 6 recebe mensagem da RSU 9 e retransmite



Fonte: Elaborada pela autora.

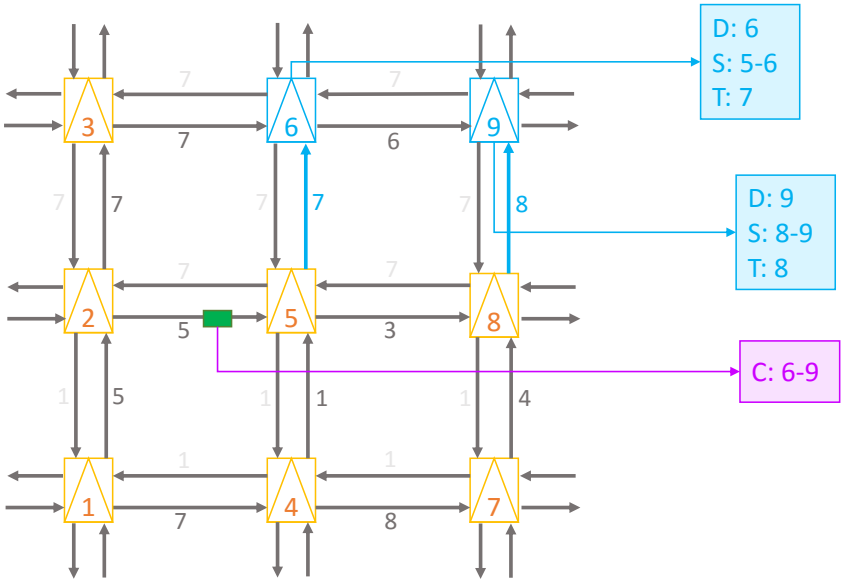
Figura 8 – Veículo envia *WantHop* à RSU 2, que responde



Fonte: Elaborada pela autora.

Na Figura 8, o veículo pede à RSU 2 o caminho até seu destino. A RSU 2 verifica sua tabela, que já foi atualizada com o novo tempo do segmento 6-9, e envia ao veículo o caminho de menor tempo naquele momento. Aconteceu de o caminho ser o mesmo que antes. O veículo segue a rota recebida. Na Figura 9, as RSUs 6 e 9 percebem mudança no tempo de segmentos que elas controlam e enviam propaganda. As outras RSUs atualizam suas tabelas e retransmitem as mensagens (omisso) com o novo tempo.

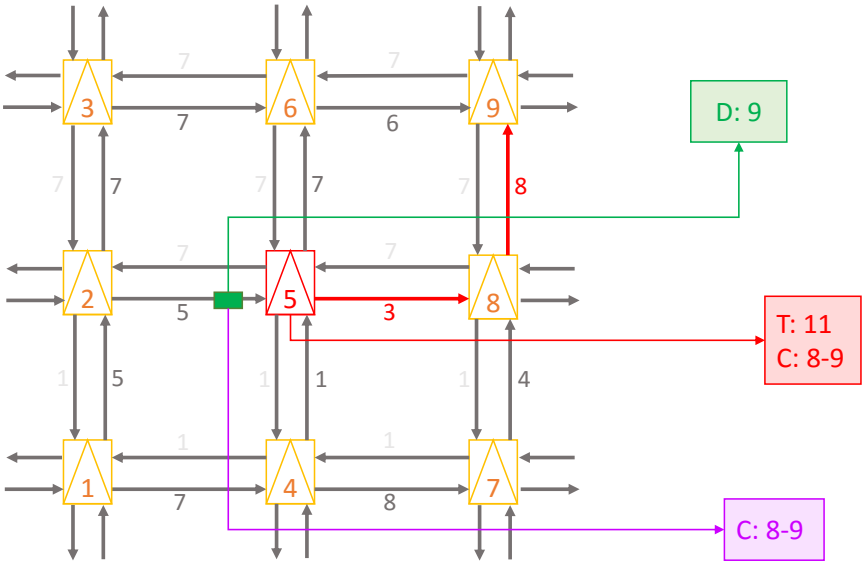
Figura 9 – RSUs 6 e 9 percebem alterações em seus segmentos



Fonte: Elaborada pela autora.

Na Figura 10, o veículo pede à RSU 5, que já atualizou sua tabela com os novos tempos, o caminho até seu destino. A RSU 5 recupera o caminho de menor tempo na sua tabela e informa ao veículo. Nesse momento, é possível perceber que o caminho original do veículo muda, pois aparece um caminho com tempo menor. Na Figura 11, o veículo continua percorrendo seu caminho e enviando mensagens de *WantHop* às RSUs, que respondem.

Figura 10 – Veículo envia *WantHop* à RSU 5, que responde

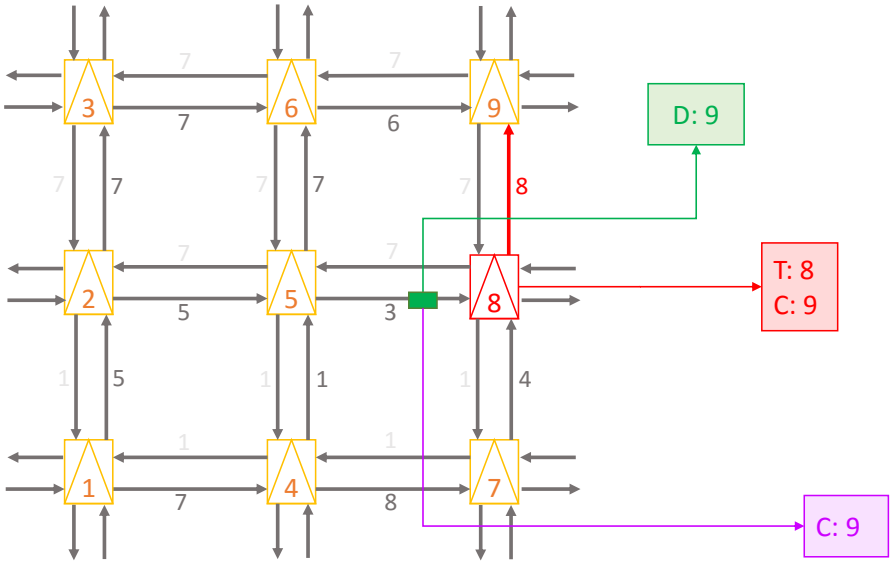


Fonte: Elaborada pela autora.

Na Figura 12, o veículo chega a um segmento do seu destino, e portanto suas mensagens de *WantHop* servem apenas para informar à RSU do destino, 9, sobre sua

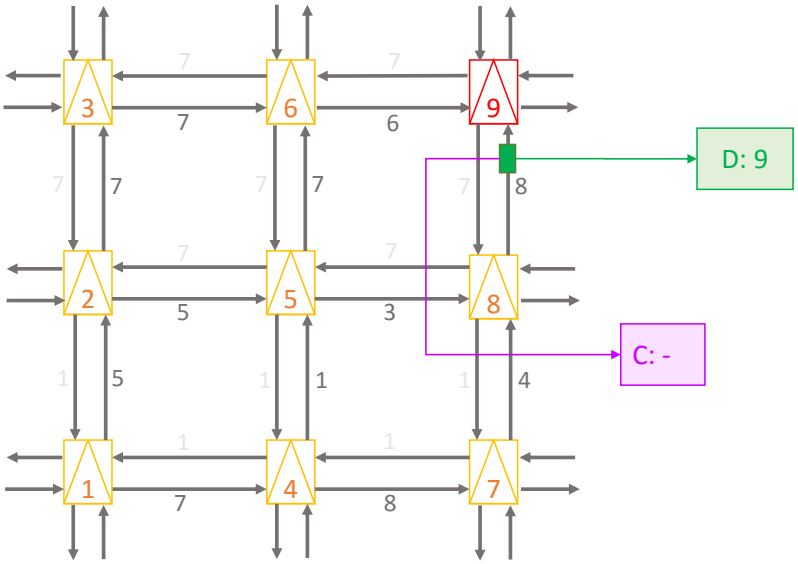
presença, para que ela possa atualizar as informações do segmento usando as informações recebidas do veículo.

Figura 11 – Veículo envia *WantHop* à RSU 8, que responde



Fonte: Elaborada pela autora.

Figura 12 – Veículo envia *WantHop* à RSU 9, apenas para alertá-la



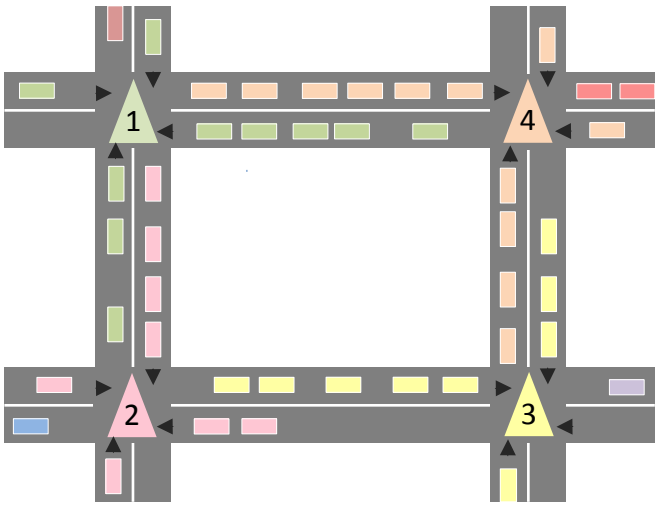
Fonte: Elaborada pela autora.

3.2 Segmentos de via

Um segmento de via é um sentido possível de uma rua ou avenida, sendo que ele inicia na interseção da qual os veículos partem e termina na interseção na qual os veículos chegam. Cada RSU é responsável por guardar as informações dos seus segmentos de via e verificar mudanças na velocidade e tempo médios para cruzar esses segmentos. No caso de

um segmento muito longo, ele será dividido em segmentos menores e RSUs serão inseridas para cuidar deles, para que não existam regiões sem controle. A divisão do segmento será em múltiplos do tamanho médio dos segmentos.

Figura 13 – RSUs e seus segmentos de via



Fonte: Elaborada pela autora.

Um segmento S é considerado como sendo da RSU R caso S termine em R, ou seja, as RSUs cuidam dos segmentos que chegam nelas. Caso uma mudança considerável seja detectada na velocidade média para percorrer algum segmento, a RSU responsável vai começar o processo de atualização da tabela de roteamento, informando às outras RSUs sobre o novo tempo do segmento que mudou. A Figura 13 ilustra algumas RSUs com seus respectivos segmentos. Os triângulos representam as RSUs e os retângulos são os veículos naquele segmento. Os veículos de cada RSU, isto é, que estão em algum segmento controlado pela RSU, foram representados da mesma cor que sua RSU.

Quadro 1 – Informações armazenadas sobre um segmento

Parâmetro	Descrição
Id	Identificador único do segmento.
RSU origem	RSU na qual o segmento inicia.
Comprimento	Comprimento do segmento, em metros.
VelMax	Velocidade máxima permitida no segmento, em m/s.
VelMed	Velocidade média do segmento, em m/s.
TmpMed	Tempo médio para um veículo percorrer o segmento, em segundos.
Faixa	Faixa de velocidade atual do segmento.

Fonte: Elaborado pela autora.

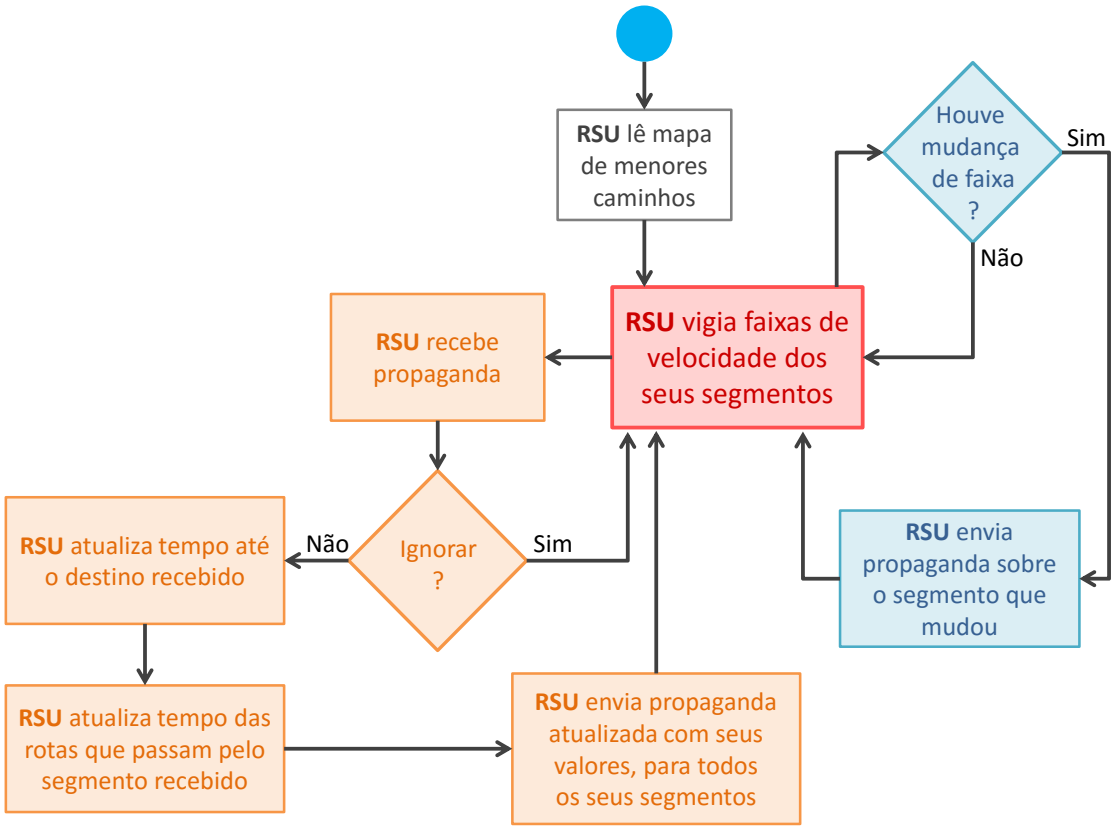
Tanto a origem como o destino de um veículo são segmentos. Como as RSUs sabem qual RSU é a controladora de cada segmento, um veículo sempre sai de uma RSU para chegar em outra. Portanto, o roteamento é realizado salto-a-salto, com os veículos saindo de uma RSU, passando por outras no caminho e chegando à RSU destino. As informações dos segmentos mantidas pelas RSUs estão descritas no Quadro 1.

O id, a RSU origem, o comprimento e a velocidade máxima são informações fixas de cada segmento. A velocidade média é calculada de acordo com as informações recebidas dos veículos daquele segmento. O tempo médio e a faixa de velocidade são calculados pela RSU usando a velocidade média.

3.3 Atualização dinâmica da tabela de roteamento

A Figura 14 ilustra o processo de atualização dinâmica da tabela de roteamento. Inicialmente, todas as RSUs sabem os caminhos de menor tempo para todas as outras, considerando que não há veículos nos segmentos e que um veículo pode cruzar os segmentos na velocidade máxima permitida, ou seja, as RSUs sabem os menores caminhos para as outras. Essa tabela inicial é fixa e é sempre lida para a memória das RSUs no início. Quando os veículos começam a se movimentar pelos segmentos, a tabela vai sendo atualizada com novos tempos para os destinos, já que as velocidades médias dos segmentos e, portanto, os tempos de viagem por esses segmentos vão mudando.

Figura 14 – Processo de atualização da tabela de roteamento

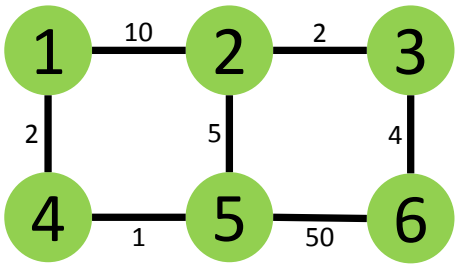


Fonte: Elaborada pela autora.

3.3.1 Tabela de roteamento

Cada RSU mantém uma tabela de roteamento, que guarda pelo menos um caminho até cada destino possível, podendo guardar até o número de segmentos que saem da RSU de caminhos para um mesmo destino. Por exemplo, se quatro segmentos saem da RSU, ela pode ter até quatro caminhos possíveis, a partir dela, para um mesmo destino.

Figura 15 – Exemplo de mapa para demonstrar a tabela



Fonte: Elaborada pela autora.

Na Figura 15 há um exemplo de mapa, com RSUs nas interseções. Todos os segmentos são de ida e volta e o número indica o tempo de viagem por aquele segmento. Na Tabela 2 há um exemplo de como seria a tabela de roteamento para o momento atual da RSU 1 da Figura 15. Na tabela, *TempoD* é o tempo total até o destino e *TempoPS* é o tempo até o próximo salto.

Tabela 2 – Exemplo de uma tabela de roteamento

	Caminho1			**Caminho2**		
Destino	**Saltos**	**TempoD**	**TempoPS**	**Saltos**	**TempoD**	**TempoPH**
2	4-5-2	8	2	2	10	10
3	4-5-2-3	10	2	2-3	12	10
4	4	2	2	2-5-4	16	10
5	4-5	3	2	2-5	15	10
6	4-5-2-3-6	14	2	2-3-6	16	10

Fonte: Elaborada pela autora.

Cada caminho guardado na tabela contém informações sobre ele mesmo, que são o destino, o tempo até o destino, a RSU do próximo salto, o tempo até o próximo salto e os quatro saltos após o próximo salto, caso existam, até o destino. Cinco saltos são guardados para que um veículo não fique sem rota caso não receba uma mensagem com o próximo salto a tempo. Ao longo desses saltos, o veículo recebe novas mensagens com a rota e atualiza o caminho que ele deve seguir.

3.3.2 Mensagens de atualização inicial controlada

Para evitar que a rede seja sobrecarregada com pacotes de atualização da tabela de roteamento, uma RSU só envia novas mensagens com informações sobre os seus segmentos (mensagem de propaganda) caso haja uma diferença significativa na velocidade média

dos veículos, ou seja, no tempo que eles levam para cruzar o segmento. No Quadro 2, é mostrada a estrutura de uma mensagem de propaganda.

Quadro 2 – Estrutura de uma mensagem de propaganda

Parâmetro	Descrição
Destino	Id da RSU do destino sendo ofertado.
Emissora	Id da RSU emissora da mensagem.
Penúltima emissora	Id da penúltima RSU emissora da mensagem.
Segmento até emissora (seg)	Segmento de via que liga na emissora, primeiro salto.
Tempo até o destino	Tempo em segundos até o destino.
Tempo até a emissora	Tempo em segundos até a RSU emissora da mensagem.
Hop2	Segundo salto.
Hop3	Terceiro salto.
Hop4	Quarto salto.
Hop5	Quinto salto.

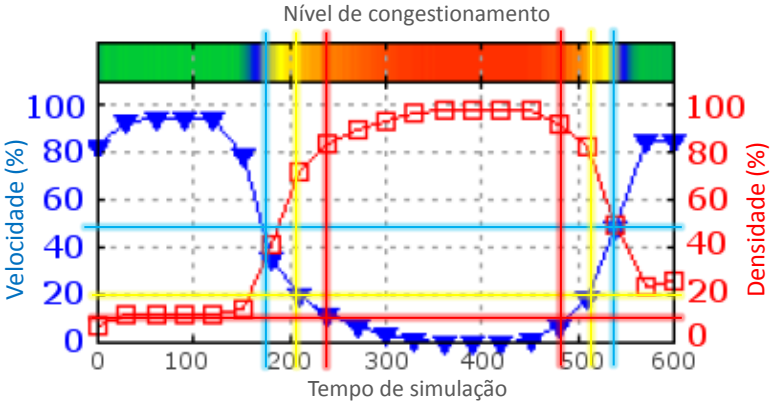
Fonte: Elaborado pela autora.

Para controlar o envio de mensagens de propaganda, quatro faixas de velocidade de um segmento foram definidas, como indicado na Figura 16. Quando algum dos seus segmentos muda de faixa de velocidade, a RSU inicia o processo de atualização, enviando uma mensagem de propaganda referente ao segmento que mudou de faixa.

Figura 16 – Faixas de velocidade de um segmento

Fonte: Elaborada pela autora.

Figura 17 – Justificativa das faixas de velocidade dos segmentos



Fonte: Adaptada de (ARAUJO, 2014).

As quatro faixas foram definidas de acordo com o nível de congestionamento estimado pelo sistema *fuzzy* proposto em (ARAUJO, 2014), como indicado pelas linhas na Figura 17. O nível de congestionamento vai de livre (verde) a severo (vermelho), passando por fraco (azul) e moderado (amarelo). É possível perceber que com 50, 20 e 10% da velocidade máxima permitida (curva azul), o congestionamento estimado pelo sistema *fuzzy* muda de nível, passando de livre/fraco para moderado, de moderado para

ficando severo, e finalmente para severo. Portanto, esses valores foram escolhidos para delimitar as faixas que controlam a atualização das tabelas de roteamento das RSUs.

$$porcVel = \frac{velMed}{velMax} \tag{3.1}$$

Uma RSU calcula em qual das faixas seus segmentos se encontram periodicamente, de acordo com a velocidade média estimada para os segmentos. A velocidade média dos segmentos é calculada a partir das mensagens recebidas dos veículos, que informam à RSU sua velocidade ao longo do segmento. Então, usando a Equação 3.1, a RSU calcula a porcentagem da velocidade atual do segmento, para decidir a faixa de velocidade. Na equação, $porcVel$ é a porcentagem da velocidade máxima do segmento, que é usada para saber a faixa, $velMed$ é velocidade média do segmento calculada no momento, e $velMax$ é velocidade máxima do segmento, que é uma informação fixa de cada segmento, armazenada pelas RSUs.

3.3.3 Retransmissão da mensagem de atualização

Uma RSU que recebe uma propaganda atualiza sua tabela com as informações recebidas, caso seja vizinha direta da RSU emissora, já que não faz sentido uma RSU sugerir um segmento que é inalcançável a partir dela para os veículos. Primeiramente, a RSU verifica se já conhece aquele destino através daquele segmento e, em caso positivo, atualiza as informações.

Quadro 3 – Mensagem de propaganda retransmitida

Parâmetro	Padrão	Retransmissão
Destino	RSU destino	RSU destino
Emissora	RSU anterior à atual	RSU atual
Penúltima	RSU anterior à anterior	RSU anterior à atual
Segmento (seg)	seg da emissora (segAnt)	seg da RSU atual (segAt)
Tempo ao destino (TmpDes)	TmpDes	TmpDes recebido + Tempo por segAt
Tempo à emissora	Tempo para passar por segAnt	Tempo para passar por segAt
Hop2	Segundo salto	segAnt
Hop3	Terceiro salto	Hop2 recebido
Hop4	Quarto salto	Hop3 recebido
Hop5	Quinto salto	Hop4 recebido

Fonte: Elaborado pela autora.

Caso conheça o destino, mas não através daquele segmento, a RSU insere um novo possível caminho com as informações recebidas na lista do destino. Em seguida, a RSU precisa retransmitir a propaganda recebida, atualizando-a com seus dados. Para cada segmento que a RSU cuida, ela vai enviar uma propaganda atualizada com os tempos de viagem por aquele segmento. No Quadro 3 é possível verificar como essa atualização é feita para a retransmissão. Para evitar que uma propaganda seja retransmitida

indefinidamente, o campo de penúltimo emissor da mensagem é usado. Quando uma RSU recebe uma propaganda cujo penúltimo emissor é ela própria, essa mensagem é descartada. Caso a RSU receba uma mensagem sobre um destino que é ela própria, a mensagem também será descartada.

3.4 Processo de roteamento

O processo de roteamento é feito salto-a-salto (*hop*-a-*hop*), para cada veículo. Começando em uma RSU origem, o veículo deve chegar a uma RSU destino e, para tanto, pede à sua RSU o próximo salto, que é o próximo segmento para o qual ele deve ir. Esse pedido de rota realizado pelos veículos é controlado por cada um deles, individualmente, com base no número de saltos armazenado e na velocidade atual do veículo, para enviar mais ou menos mensagens de acordo com a necessidade.

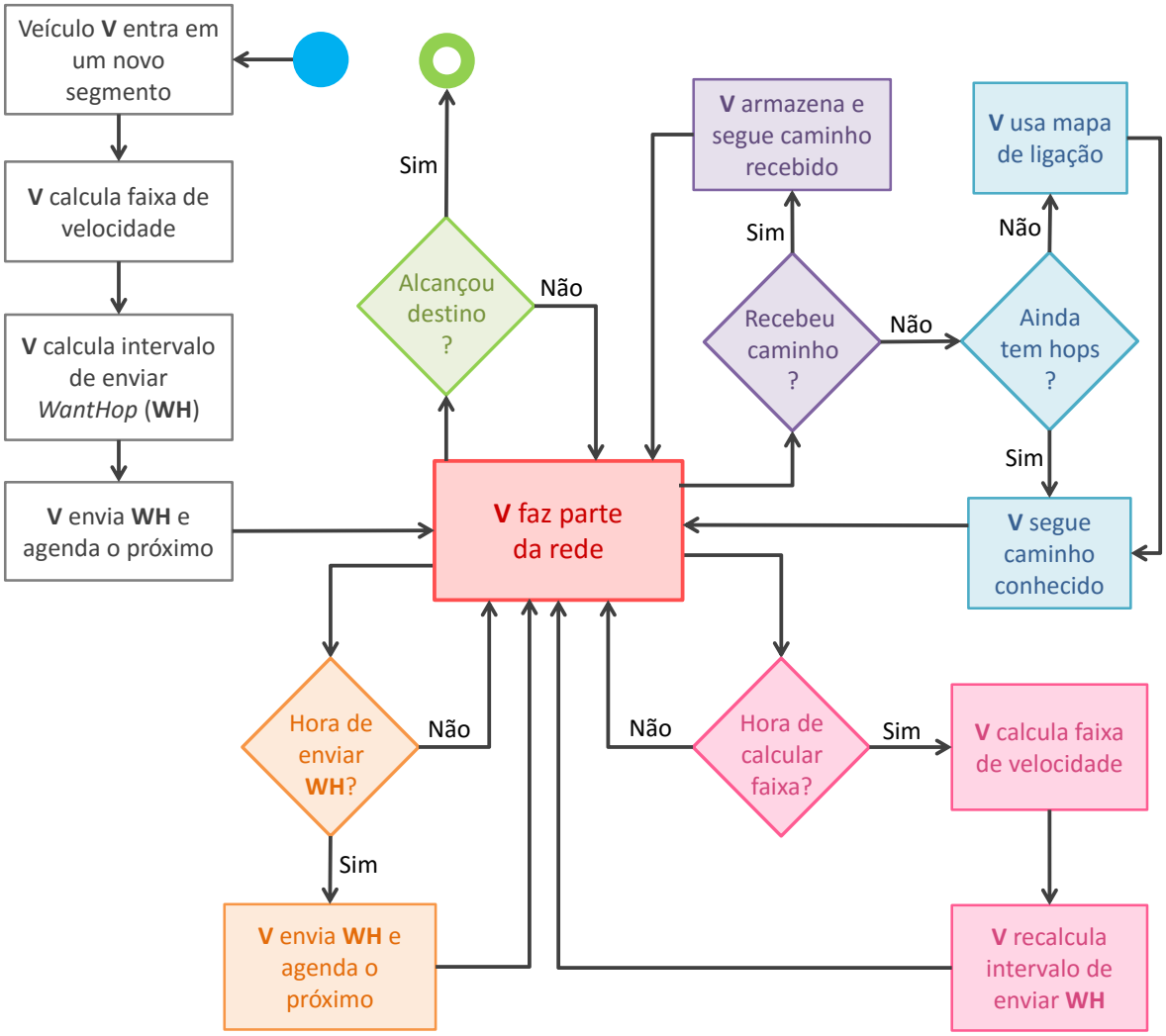
Uma RSU sabe que um veículo pertence à área controlada por ela, pois o veículo informa, em suas mensagens, em qual segmento ele está no momento. A RSU, então, verifica, na sua tabela de roteamento, os caminhos possíveis para aquele destino e informa a todos os veículos do segmento o caminho que some o menor tempo total até o destino, naquele momento. A resposta da RSU pode conter até cinco segmentos que levam o veículo ao seu destino. O número de segmentos informados depende de o segmento do destino já estar na resposta ou não. Caso o segmento do destino esteja na resposta, menos que cinco segmentos podem ser informados. A seguir são detalhados o pedido de rota realizado pelos veículos e a resposta das RSUs.

3.4.1 Pedido de rota controlado

Na Figura 18, está ilustrado o processo de roteamento que acontece nos veículos seguindo o D-Hop. Imediatamente após entrar em um novo segmento, um veículo V envia um pedido de novo salto (*WantHop*) para a RSU à qual o segmento que ele está pertence (estrutura da mensagem no Quadro 4) e espera uma resposta por um intervalo de tempo dinâmico, controlado pelo próprio veículo usando suas informações.

Quanto menos saltos V conhece e mais rápida é sua velocidade, mais frequente é seu envio, para tentar garantir que ele consiga receber uma nova rota antes de ficar sem caminho. Caso não receba a mensagem de próximo salto após o intervalo de tempo, V reenvia a mensagem de pedido e recomeça a contagem do intervalo. Juntamente com o pedido de salto, V informa sua velocidade à RSU, para que ela possa calcular a velocidade média do segmento.

Figura 18 – Processo de roteamento nos veículos



Fonte: Elaborada pela autora.

Quadro 4 – Mensagem de pedido de próximo salto (WantHop)

Parâmetro	Descrição
Id	Id do veículo emissor.
Destino	Id da RSU do destino desejado.
Segmento atual	Segmento no qual o veículo se encontra.
Velocidade	Velocidade do veículo (m/s).

Fonte: Elaborado pela autora.

O intervalo de tempo para pedido de próxima rota (iWH) é dado em segundos e calculado de acordo com a Equação 3.2. Ele depende do tempo médio (TM) que um veículo leva para cruzar um segmento na velocidade máxima permitida, do número de saltos que o veículo ainda tem armazenado, e da faixa de velocidade em que o veículo se encontra.

$$iWH(s) = TM(s) \times porcNS \times fatorDes \tag{3.2}$$

Na equação, o tempo médio (TM) é calculado de acordo com a Equação 3.3. A porcentagem do número de saltos ($porcNS$) é calculada de acordo com o Quadro 3, e leva em consideração o número de saltos que o veículo ainda sabe para ajustar sua frequência de pedido de novo salto. Quanto menos saltos, mais frequente. O fator de desaceleração ($fatorDes$) é definido de acordo com a faixa de velocidade que o veículo se encontra, como indicado no Quadro 4, e serve para diminuir a frequência de pedido de novo salto caso o veículo não esteja na velocidade máxima. A seguir, serão explicados em mais detalhes cada um dos termos da equação: (i) o tempo médio, (ii) a porcentagem do número de saltos, e (iii) o fator de desaceleração.

Tempo médio (TM):

Na Equação 3.3, o tempo médio (TM) é dado em segundos. O parâmetro $tamSeg$ indica o tamanho médio de segmento do mapa, ignorando segmentos muito longos ou muito curtos, em metros. O parâmetro $velMax$ indica a velocidade máxima permitida média do mapa, também ignorando velocidades muito altas ou muito baixas, em metros por segundo.

$$TM(s) = \frac{tamSeg(m)}{velMax(m/s)} \tag{3.3}$$

Para os dois parâmetros, foram ignorados valores muito altos ou muito baixos para que o tempo médio não ficasse desregulado devido a segmentos que são casos especiais. Usar o cálculo do tempo médio dessa forma garante que o intervalo de pedido de nova rota esteja de acordo com o mapa em uso e, também, que os veículos não precisem saber informações não pertinentes a eles, como o tamanho de cada segmento do mapa.

Porcentagem do número de saltos (porcNS):

Uma das funções da mensagem *WantHop* é avisar às RSUs sobre a presença de veículos em seus segmentos, informando suas velocidades. Portanto, todos os veículos enviam um *WantHop* assim que entram em um novo segmento. Como pelo menos um *WantHop* tem seu envio garantido, é possível diminuir a frequência de envio dessas

mensagens, para garantir que a rede não seja sobrecarregada de pacotes. A diminuição da frequência é feita de acordo com o número de saltos que um veículo ainda sabe, como é possível observar na Tabela 3. Se o veículo conhece o salto imediato que o leva ao destino ou quanto mais saltos ele conhece, menos frequente pode ser seu envio de mensagens *WantHop*, pois, nesses casos, elas servem mais pra alertar à RSU sobre a presença do veículo do que para pedir novos saltos.

Tabela 3 – Frequência dinâmica de pedido de próxima rota

Número de saltos conhecidos	Frequência de envio	porcNS
Conhece o destino	100% do TM	1
5	100% do TM	1
4	90% do TM	0.9
3	80% do TM	0.8
2	70% do TM	0.7
1	50% do TM	0.5

Fonte: Elaborada pela autora.

Caso um veículo V tenha recebido seu destino como um próximo salto, V muda sua frequência de envio para 100% do tempo médio, permanentemente, pois agora suas mensagens servem apenas para alertar às RSUs sobre sua presença nos segmentos. Se o veículo ainda não tem seu destino na lista de próximos saltos, as porcentagens são ajustadas de acordo com o número de saltos conhecidos. No caso de um veículo V só conhecer o próximo salto, a porcentagem será de 50% do tempo médio, ou seja, V enviará pelo menos duas ou três mensagens de *WantHop* à RSU enquanto estiver no segmento dela, para garantir que receba uma resposta a tempo e não fique sem rota. As porcentagens foram decididas de forma empírica, a partir de várias simulações, para garantir que todos os veículos sempre tivessem um próximo salto com o menor número possível de pedidos.

Fator de desaceleração (fatorDes):

Para controlar ainda mais o envio das mensagens *WantHop*, foram criadas faixas de velocidade para os veículos, indicadas na Figura 19. Um veículo V calcula em qual faixa de velocidade ele está de minuto em minuto, através da Equação 3.4, onde $porcFV$ indica a porcentagem da velocidade e é usada para saber a faixa, vel indica a velocidade do veículo no momento do cálculo, e $velMax$ indica a velocidade máxima permitida no segmento que o veículo se encontra.

Figura 19 – Faixas de velocidade de um veículo

Fonte: Elaborada pela autora.

$$porcFV = \frac{vel}{velMax} \tag{3.4}$$

O fator de desaceleração ($fatorDes$), calculado através da Equação 3.5, expressa o aumento causado no tempo que um veículo leva para percorrer um segmento em comparação com o tempo médio (TM), causado pela diminuição da velocidade do veículo de acordo com as porcentagens do limite superior das faixas de velocidade. A diminuição da velocidade está expressa na parte onde $velMax$ (velocidade máxima permitida média do mapa) é multiplicada por $porcFV$ (limite superior da faixa do fator de desaceleração sendo calculado).

$$fatorDes = \frac{\frac{tamSeg}{velMax \times porcFV}}{TM} = \frac{TM \times \frac{1}{porcFV}}{TM} = \frac{1}{porcFV} \tag{3.5}$$

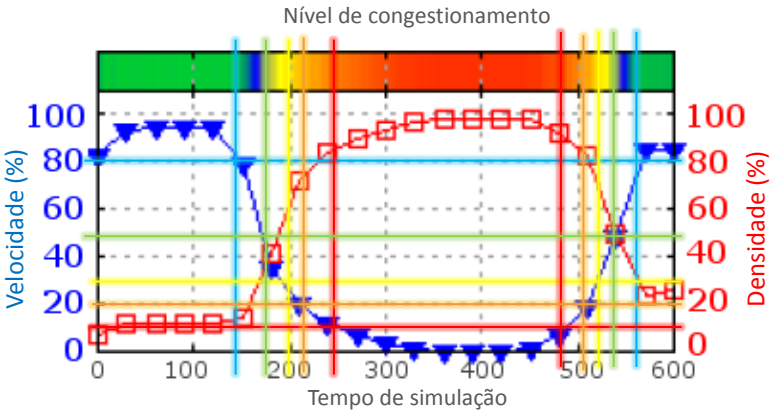
Tabela 4 – Fatores de desaceleração das faixas de velocidade

Faixa	porcFV	fatorDes
1 (100-80%)	100%	1
2 (80-50%)	80%	1.25
3 (50-30%)	50%	2
4 (30-20%)	30%	3.3
5 (20-10%)	20%	5
6 (10-0%)	10%	10

Fonte: Elaborada pela autora.

Assim, é possível montar a Tabela 4 com os valores dos fatores que devem ser usados no cálculo do intervalo de pedido de novo salto. Com o uso dos fatores, é possível aumentar ou diminuir a frequência do envio das mensagens *WantHop* de acordo com a velocidade dos veículos sem que eles precisem ter conhecimento do tamanho dos segmentos, já que isso é abstraído no cálculo do fator. Isso também torna possível que o cálculo do intervalo continue dinâmico, independente do mapa.

Figura 20 – Justificativa das faixas de velocidade dos veículos



Fonte: Adaptado de (ARAUJO, 2014).

Da mesma forma que as faixas de velocidade de um segmento foram definidas para as RSUs, as faixas de velocidade para os veículos foram definidas de acordo com o nível de congestionamento estimado pelo sistema *fuzzy* proposto em (ARAUJO, 2014), como

indicado pelas linhas na Figura 20. Novamente, o nível de congestionamento vai de livre (verde) a severo (vermelho), passando por fraco (azul) e moderado (amarelo).

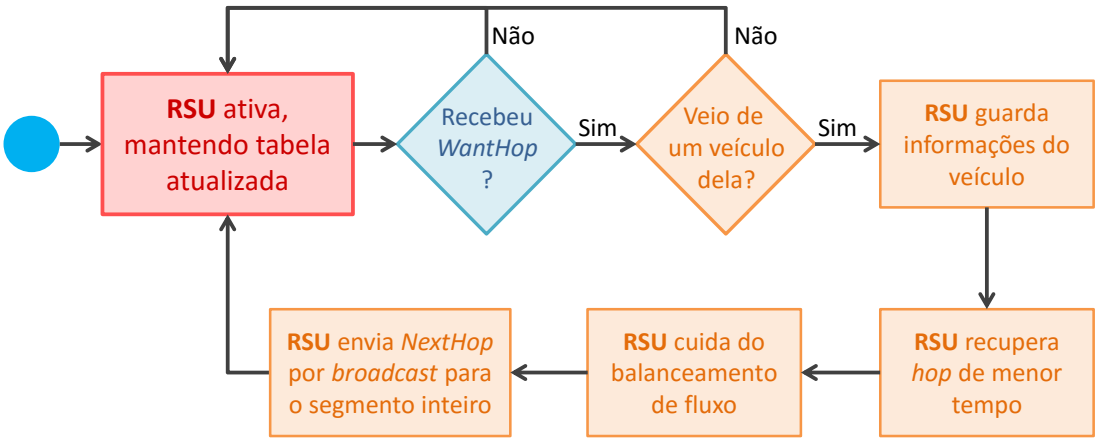
É possível perceber que com 80, 50, 30, 20 e 10% da velocidade máxima permitida (curva azul), o congestionamento estimado pelo sistema *fuzzy* muda de nível. Para os veículos, foram definidas mais faixas que para os segmentos, pois, assim, é possível um controle maior sobre o envio de mensagens, visto que há uma atenção maior para as mudanças de velocidade. Principalmente com 20 e 10% da velocidade, que, mesmo não havendo uma grande diferença no nível de congestionamento, o fator de desaceleração cresce consideravelmente mais, pois os veículos estão quase parados.

Dessa forma, o intervalo de pedido de novo salto é dinâmico e controlado para cada segmento, número de saltos e velocidade que os veículos possam apresentar. O controle no envio das mensagens de *WantHop* garante que a rede não seja sobrecarregada com pacotes no caso de regiões com velocidades diminuídas e, portanto, tempos maiores que o tempo médio para os veículos cruzarem essas regiões.

3.4.2 Resposta de rota

A mensagem de resposta da RSU (*NextHop*, Quadro 5) com o próximo salto, que soma o menor tempo no momento, é enviada por *broadcast* para os veículos presentes no segmento em questão, assim que a RSU recebe um pedido de próximo salto (*WantHop*). Todos os veículos que estavam esperando o próximo salto para o destino informado salvam os saltos recebidos e se dirigem para o caminho informado. Na Figura 21, está ilustrado o processo realizado pelas RSUs em resposta ao pedido de rota.

Figura 21 – Resposta de rota



Fonte: Elaborada pela autora.

As RSUs só respondem a pedidos de rota de veículos que estejam nos seus segmentos. Caso o pedido venha de um veículo da RSU, ela armazena as informações do veículo, recupera o caminho de menor tempo realizando um balanceamento do tráfego

e envia a mensagem de resposta para o segmento inteiro de onde veio o pedido. No Quadro 5, está a estrutura da mensagem de resposta de rota.

Quadro 5 – Mensagem de resposta de próximo salto (NextHop)

Parâmetro	Descrição
Id	Id da RSU emissora.
Destino	Id da RSU do destino desejado.
Próximo salto	Segmento do próximo salto, que é o primeiro salto.
Segmento	Segmento do qual foi pedido o próximo salto.
Hop2	Segundo salto.
Hop3	Terceiro salto.
Hop4	Quarto salto.
Hop5	Quinto salto.

Fonte: Elaborado pela autora.

O balanceamento é feito para prevenir que o segmento de menor tempo até o destino acabe ficando lento, devido a todos os veículos serem roteados para ele. A RSU envia os veículos para segmentos de saída diferentes desde que o tempo até o destino esteja dentro de um limite, para não prejudicar o tempo de viagem. Os segmentos disponíveis para a RSU direcionar os veículos são todos os que saem dela, exceto o que mandaria o veículo de volta para onde ele veio. Isto é, se o veículo veio da RSU 1 para a RSU 2, e existe o caminho da RSU 2 para a 1, a RSU 2 não vai considerar mandar o veículo de volta para a 1, pois isso faz com que o veículo fique preso entre essas duas RSUs, indo e voltando até outro caminho ser o de menor tempo. Apenas caso esse segmento de retorno seja o único disponível, ele será usado.

A partir dos segmentos disponíveis para enviar os veículos, a RSU vai descobrir o de menor tempo e, caso o de menor tempo seja o último que ela enviou veículos para, ela vai verificar os outros. Se o tempo até o destino por algum dos outros caminhos disponíveis não ultrapassar um minuto do caminho de menor tempo, a RSU vai enviar os veículos para esse caminho alternativo, salvando-o como último segmento para o qual enviou veículos.

3.5 Análise de complexidade

Nos veículos seguindo o D-Hop, todo o processamento que eles fazem é $\theta(1)$, pois consiste apenas em atualizar os intervalos de pedir nova rota de acordo com o número de saltos conhecidos e velocidade atual, guardar a nova rota recebida e seguir a rota armazenada. Nas RSUs, existe o custo de atualizar a tabela de roteamento quando uma mensagem de atualização é recebida, o custo de calcular a faixa de velocidade atual dos segmentos e o custo de recuperar o menor caminho até certo destino quando a RSU recebe um pedido de rota. Fora esses três, os outros custos são $\theta(1)$, que são enviar mensagem de atualização da tabela caso algum segmento tenha mudado de faixa de velocidade e enviar

a resposta de rota após recuperar o caminho de menor tempo. Os custos que não são $\theta(1)$ estão descritos a seguir.

3.5.1 Atualização da tabela de roteamento

Assim que a RSU recebe uma mensagem para atualizar a tabela de roteamento, ela verifica se essa mensagem deve ser ignorada. Caso não, a tabela de roteamento será atualizada. Inicialmente a RSU vai verificar se ela conhece o destino informado e, caso não, a informação do novo destino será inserida no mapa em $\theta(1)$. Caso o destino seja conhecido, a RSU verifica se já tem uma entrada para o caminho recebido em $\theta(H)$, sendo H o número de caminhos. Caso não tenha uma entrada para o caminho recebido, será inserido um novo possível caminho para o destino em $\theta(1)$. Caso a RSU tenha uma entrada para o caminho recebido, suas informações serão atualizadas em $\theta(1)$. Portanto, o custo para atualizar a entrada do destino é $\theta(H)$.

Assim que a RSU atualiza a entrada do destino, ela vai procurar outros caminhos onde o emissor da mensagem de atualização é o próximo salto e atualizar o tempo até o destino por esses caminhos usando as informações recebidas. Para isso, a RSU vai olhar todos os destinos possíveis, que no caso é o número de RSUs, R, recuperando a lista de caminhos para cada destino em $\theta(1)$. Para cada caminho, caso o próximo salto seja a RSU emissora da mensagem de atualização, a informação de tempo até o destino será atualizada com o novo tempo até o próximo salto. No pior caso, o custo da atualização para uma entrada de destino será $\theta(1)$, pois não é armazenado mais de um caminho para o mesmo destino que passe pelo mesmo segmento. O custo para atualizar os caminhos que envolvem a RSU emissora da mensagem é, portanto, linear no número de destinos possíveis: $\theta(R)$.

Finalmente, a RSU vai enviar uma mensagem de atualização da tabela para cada segmento dela, informando às outras RSUs sobre as novas informações até aquele destino passando pelos seus segmentos. O custo desse envio é $\theta(S)$, que é linear no número de segmentos da RSU.

3.5.2 Cálculo da faixa de velocidade do segmento

Em intervalos fixos, a RSU vai calcular a faixa de velocidade dos seus segmentos. Para cada segmento, num total de S segmentos, a RSU vai calcular a média da velocidade apresentada pelos V veículos. No total o custo é $\theta(S * V)$, que é quadrático, mas nunca muito grande, pois S é o número de segmentos chegando numa interseção e V é o número de veículos em um único segmento.

3.5.3 Recuperação do caminho de menor tempo

Para recuperar o caminho de menor tempo e cuidar do balanceamento, a RSU começa recuperando a lista de possíveis próximos saltos até o destino desejado, em $\theta(1)$, pois é implementado como um mapa. Então, a RSU vai verificar essa lista de saltos uma vez para recuperar o de menor tempo em $\theta(H)$, que é o tamanho da lista. Conhecendo o caminho de menor tempo até o destino, a RSU verifica se é o mesmo caminho que ela enviou pela última vez. Caso sim, para realizar o balanceamento proposto, a RSU verifica a lista de saltos novamente em $\theta(H)$, procurando outro caminho que esteja dentro do intervalo de tempo aceitável. Portanto, no pior caso o custo de recuperar o caminho de menor tempo é $\theta(H)$.

4 RESULTADOS

4.1 Configurações do ambiente de simulação

Para a avaliação do D-Hop, foi criada uma rede veicular simulada. Os simuladores usados foram o SUMO e o OMNeT++, que, como mencionado anteriormente, são bastante usados para esse propósito em conjunto com o Veins. As simulações foram realizadas em seis máquinas virtuais executando Ubuntu 12.04, com 14 GB de memória RAM e 8 núcleos, todas no serviço de nuvem Azure, da Microsoft, o que tornou possível executar as simulações em paralelo, diminuindo o tempo necessário. As versões dos simuladores usadas foram a 4.3 do OMNeT++, a 0.15.0 do SUMO, e a versão 2.2 do Veins/MiXiM.

Todos os cenários simulados foram sobre um *Manhattan Grid* 5×5, com RSUs em todas as interseções, inclusive as externas. Cada via foi simulada com 200 metros e todas tinham os dois sentidos possíveis. A velocidade máxima permitida foi de 50 km/h. O alcance de comunicação das RSUs foi de 200 metros e, dos veículos, 100 metros. O tempo de simulação para todos os cenários foi de uma hora cada.

4.2 Medidas tomadas para a simulação

Usando os simuladores escolhidos para a realização deste trabalho, foi notado um problema com relação ao uso do roteamento. Os cálculos e decisões de rotear os veículos são feitos no simulador de rede, OMNeT++, e a ordem de mudar os veículos é enviada ao simulador de mobilidade, SUMO, que muda os veículos de acordo com o recebido. A partir do momento que um veículo é roteado pela primeira vez, sua rota original no SUMO é esquecida e ele fica completamente dependente de sempre ter o caminho quando precisar andar.

Quadro 6 – Exemplo do mapa de ligação

Rua/avenida atual	Próxima rua/avenida
Rua1	Rua2 (frente)
Rua2	Rua3 (frente)
Rua3	Rua8 (retorno)
Rua8	Rua7 (direita)

Fonte: Elaborado pela autora.

Como alguns veículos não conseguiam receber o caminho a tempo, não era enviada uma nova ordem de caminho para esses veículos ao SUMO, que os retirava da simulação. Para evitar que veículos saíssem da simulação pela falta de caminho, foram criados mapas de ligação para os cenários testados. Um mapa de ligação contém todas as ruas e avenidas

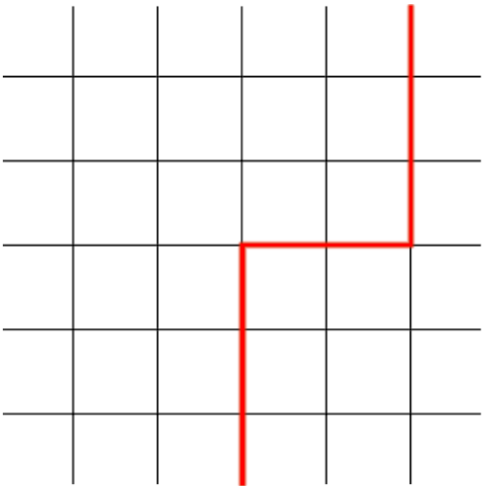
que um veículo pode estar e a rua/avenida diretamente à frente delas, caso exista. Caso não exista a rua diretamente à frente, no mapa de ligação terá ou a rua da direita, ou a da esquerda ou a de retorno, respectivamente. Apenas no caso de rua sem saída não há a próxima rua. No Quadro 6, está um exemplo de um mapa de ligação.

Um veículo que não receber o novo caminho a tempo vai usar o mapa de ligação para continuar na simulação, seguindo as vias indicadas pelo mapa enquanto não receber caminho de alguma RSU. Eventualmente, esse veículo vai conseguir receber instruções de caminho de uma das infraestruturas fixas e poderá voltar a se dirigir ao destino.

4.3 Versões e cenários simulados

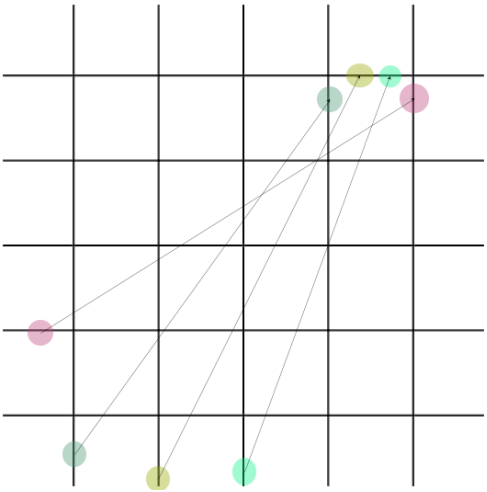
Os cenários implementados e simulados foram: (i) fluxo leve, com 200 veículos; (ii) fluxo moderado, com 400 veículos; (iii) fluxo intenso, com 1200 veículos; e (iv) fluxo único, com 720 veículos com a mesma origem e mesmo destino. O cenário de fluxo único está ilustrado na Figura 22, os outros estão ilustrados na Figura 23. Os fluxos começam na parte de baixo do mapa e sobem.

Figura 22 – Fluxo único



Fonte: Elaborada pela autora.

Figura 23 – Fluxos variados



Fonte: Elaborada pela autora.

Para todos os cenários, foram testadas versões com e sem acidentes. O cenário de fluxo leve foi testado sem acidentes (**Leve$_s$**) e com quatro acidentes (**Leve$_4$**) ao longo do tempo. O cenário moderado foi testado sem acidentes (**Médio$_s$**), com quatro (**Médio$_4$**) e com oito (**Médio$_8$**) acidentes.

O cenário intenso foi testado sem (**Intenso$_s$**), com quatro (**Intenso$_4$**) e com dezesseis (**Intenso$_{16}$**) acidentes. O cenário de fluxo único foi testado sem (**Único$_s$**) e com quatro (**Único$_4$**) acidentes. Cinco variações do D-Hop foram implementadas, para avaliar, separadamente, diferentes aspectos do mesmo, e estão descritas a seguir:

a) sem atualização (**SAtu**) - sem atualização da tabela de roteamento. Os veículos pedem a rota e as RSUs respondem, mas nunca atualizam as informações na tabela, mantendo apenas o menor caminho, que é lido para a memória no início. Esta versão foi feita para testar a diferença entre não usar um método de roteamento e usar o D-Hop;

b) sem balanceamento (**SBlnc**) - versão sem o balanceamento feito pelas RSUs de enviar os veículos cada hora para uma via de saída diferente, caso a diferença no tempo até o destino seja aceitável. Esta versão foi feita para testar o balanceamento de fluxo proposto;

c) atualização periódica (**AtPer**) - as RSUs enviam a mensagem de atualização da tabela de roteamento de forma periódica, de dois em dois minutos, em vez de apenas caso algum segmento tenha mudado de faixa de velocidade. Esta versão foi feita para comparar o controle de atualização da tabela proposto para o D-Hop com relação a nenhum controle realizado;

d) sem controle (**SCtrl**) - os veículos desta versão não controlam o envio de mensagens de pedido de rota usando sua velocidade e número conhecido de saltos, em vez disso enviam o pedido periodicamente de 10 em 10 segundos. Esta versão foi feita para comparar a diferença entre usar o controle de pedido de rota realizado pelos veículos do D-Hop e não usar controle algum;

e) ICODE adaptado, aqui denominado Um hop (**1Hop**) - esta versão é uma aproximação do ICODE, descrito anteriormente, com adaptações realizadas para ser possível simular com 45 destinos possíveis. Em comparação com o D-Hop, nesta versão os veículos pedem nova rota de 8 em 8 segundos, a atualização da tabela é periódica de quatro em quatro minutos, e as RSUs informam apenas o próximo salto para os veículos, em vez de cinco.

A versão adaptada do ICODE precisou sofrer modificações em comparação com o ICODE da literatura para ser possível testar com os 45 destinos possíveis. No original, os autores testaram apenas três destinos possíveis e, portanto, deixaram as RSUs cuidarem de anunciar o caminho para esses destinos, de forma periódica. No caso de 45 destinos possíveis, a rede seria totalmente ocupada com as mensagens periódicas.

Como apenas um salto de cada vez é enviado para os veículos, eles pedem nova rota em intervalos fixos de 8 segundos, para tentar conseguir uma resposta antes de ficar sem rota. Como no ICODE original, foi mantida a atualização periódica da tabela de roteamento, de quatro em quatro minutos para que as RSUs não fiquem ocupadas demais com a atualização e acabem não respondendo aos pedidos de rota dos veículos.

4.4 Métricas avaliadas

Algumas métricas foram escolhidas de forma a avaliar o D-Hop e compará-lo às cinco versões implementadas. O tempo de viagem e a emissão de gás carbônico (CO_2) médios, que são métricas relacionadas, foram analisadas para verificar o impacto de usar o método de roteamento proposto, no tempo de viagem e na poluição ambiental, causada pela emissão do CO_2 pelos veículos. A extensão média percorrida pelos veículos foi escolhida para analisar a diferença na distância percorrida utilizando cada versão.

A porcentagem de veículos que precisaram usar o mapa de ligação, ou seja, que ficaram sem rota em algum momento, foi calculada para analisar a eficácia do pedido de rota controlado e analisar a diferença de enviar mais de um salto de cada vez aos veículos. Além disso, o número absoluto de vezes que os veículos precisaram usar o mapa de ligação também foi analisado, para verificar quão rápido os veículos conseguiam receber rota, novamente, das RSUs. A porcentagem de veículos que alcançou o destino foi calculada para verificar o impacto do uso do mapa de ligação, pois permite saber quantos veículos conseguiram alcançar o destino antes de acabar o tempo de simulação.

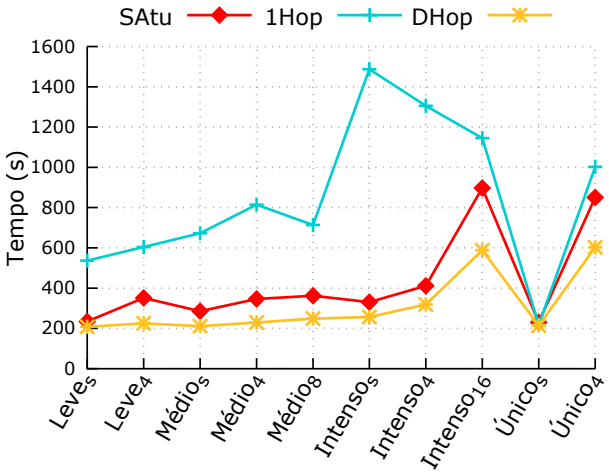
Com relação aos dois controles de envio de mensagens propostos no D-Hop, no caso das mensagens enviadas pelos veículos, o pedido de rota controlado, foram coletadas e analisadas a média de mensagens de pedido de rota enviadas por veículo e o total de mensagens enviadas durante a simulação. Para avaliar o controle da atualização da tabela de roteamento, foram coletados o número médio de mensagens de atualização da tabela enviadas por cada RSU e o total de mensagens de atualização enviadas durante a simulação.

4.5 Análise

Na Figura 24, estão os tempos médios de viagem dos veículos para cada cenário e, na Figura 25, estão as emissões de gás carbônico (CO_2) médias, das versões D-Hop, sem atualização da tabela de roteamento (SAtu), e adaptação do ICODE, 1Hop. Essas duas métricas são equivalentes, pois quanto mais tempo os veículos demoram para alcançar o destino, maior a sua emissão de CO_2.

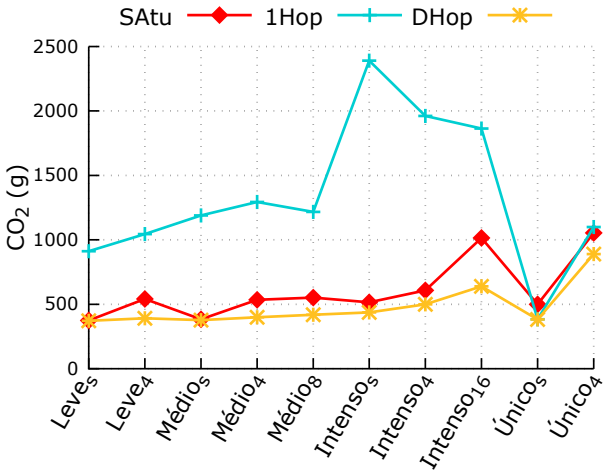
Como é possível observar, o D-Hop foi sempre melhor que as outras versões, exceto no caso do cenário único sem acidentes, no qual todas as versões apresentaram um tempo aproximado. Isso acontece, nesse cenário, pois os veículos conseguem passar pelo caminho único em velocidades próximas da máxima (vide Figura 26) e, portanto, as RSUs quase não desviam os veículos, visto que não há muitas mudanças nas faixas de velocidade dos segmentos.

Figura 24 – Tempo de viagem



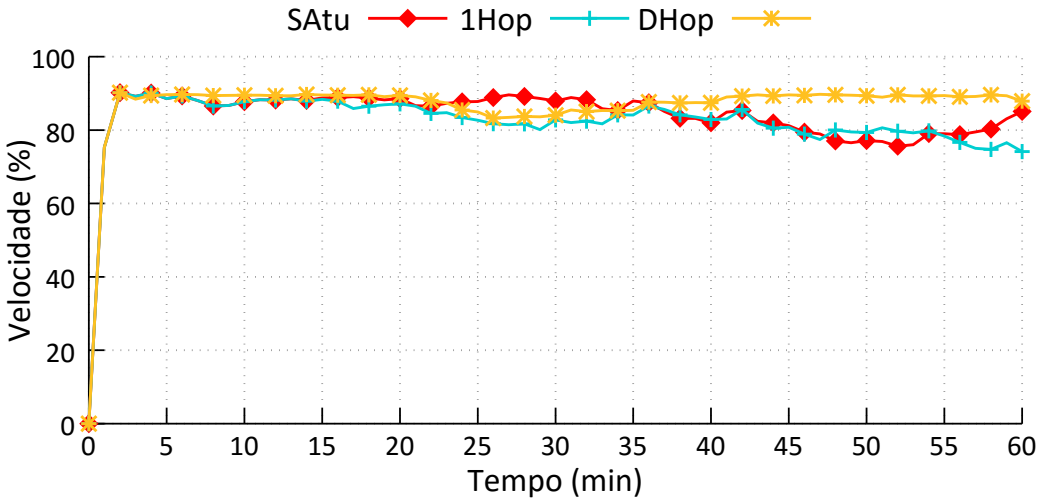
Fonte: Resultados da pesquisa.

Figura 25 – Emissão de gás carbônico



Fonte: Resultados da pesquisa.

Figura 26 – Velocidades no tempo do cenário único sem acidentes (%)



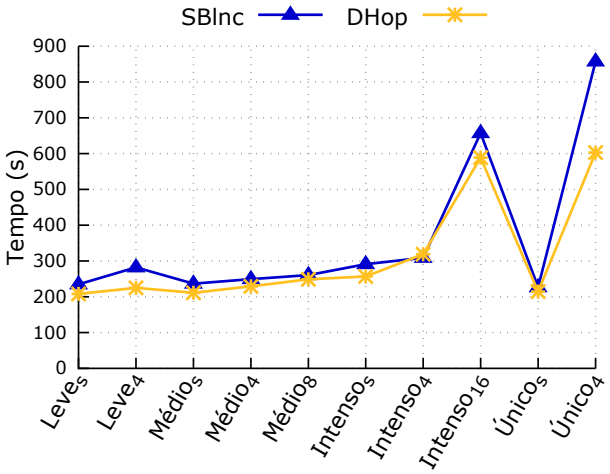
Fonte: Resultados da pesquisa.

Com relação à versão sem atualização da tabela, o D-Hop apresentou tempos menores de viagem, devido ao roteamento realizado pelas RSUs quando os segmentos começam a apresentar velocidades lentas. Sem a atualização da tabela, as RSUs continuam mandando os veículos para o menor caminho, que é o caminho conhecido inicialmente, e isso faz com que congestionamentos sejam criados nos segmentos desses caminhos, aumentando o tempo de viagem e a emissão de CO_2 dos veículos.

A versão 1Hop foi a que apresentou os piores resultados, por causa da alta porcentagem de veículos que ficam sem rota devido aos veículos conhecerem apenas o próximo salto em vez de cinco. Como decidido, sempre que um veículo fica sem rota durante a simulação, ele utiliza o mapa de ligação, e continua pedindo rota. Como os veículos usando a versão 1Hop continuavam se perdendo e usando o mapa de ligação, demorando para alcançar o destino, o tempo de viagem e a emissão de CO_2 foram aumentando, sendo o pior caso o cenário intenso sem acidentes ($Intenso_s$) com um tempo

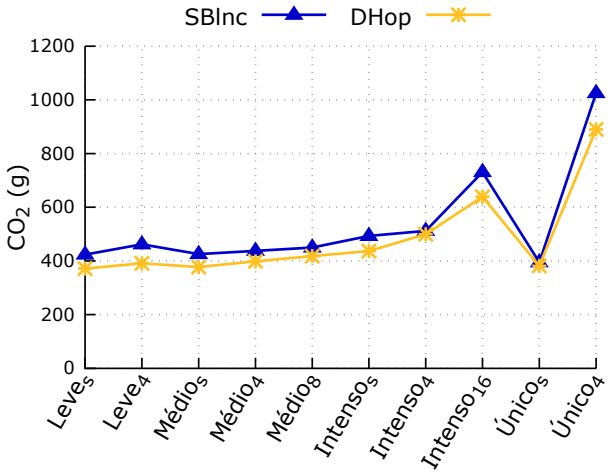
de viagem quase seis vezes maior que o D-Hop no mesmo cenário. Com o aumento do número de acidentes, os veículos precisam usar menos o mapa de ligações, já que, lentos por causa dos acidentes, eles conseguem mais respostas de rota, e portanto o tempo até o destino diminui.

Figura 27 – Tempo de viagem (SBlnc x D-Hop)



Fonte: Resultados da pesquisa.

Figura 28 – Emissão de CO_2 (SBlnc x D-Hop)



Fonte: Resultados da pesquisa.

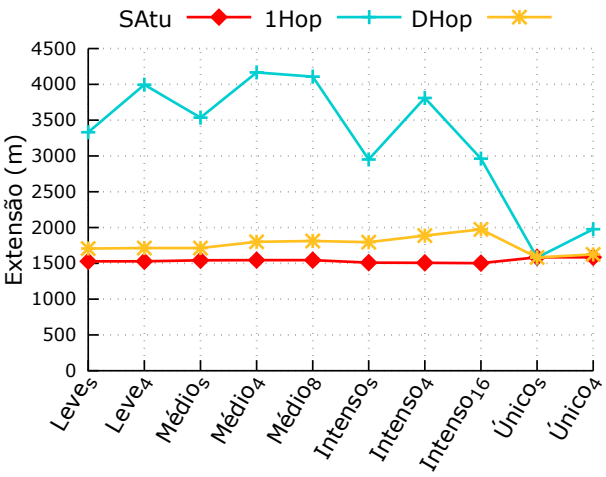
A versão sem balanceamento foi até 31% pior que o D-Hop (Figuras 27 e 28), pois, como as RSUs continuaram encaminhando os veículos para o mesmo segmento de saída sempre que ele era o de menor tempo até o destino, o mesmo tendia a congestionamento. Até a RSU controladora do segmento destino perceber a variação na faixa de velocidade do segmento e enviar a mensagem de atualização da tabela com o novo tempo do segmento, o tempo e o CO_2 totais acabam aumentando. A decisão proposta no D-Hop, de balancear os veículos entre os segmentos de saída, caso o tempo até o destino esteja dentro do limite, tem um impacto positivo na diminuição do tempo de viagem dos veículos no geral.

Na Figura 29 estão as extensões médias percorridas pelos veículos até seus destinos, para as versões sem atualização, 1Hop e D-Hop. Nos cenários de fluxo leve, não houve muita necessidade de espalhar mais os veículos para garantir tempos de viagem menores. Portanto, o D-Hop e a versão sem atualização apresentaram distâncias aproximadas. A diferença entre essas duas versões fica mais clara nos cenários de fluxo médio e de fluxo intenso, nos quais a distância média percorrida pelos veículos que seguem o roteamento proposto no D-Hop é maior. Isso ocorre porque uma distribuição maior do fluxo pelo mapa é necessária para garantir o tempo de viagem menor.

Novamente, a versão 1Hop apresentou resultados piores que as outras versões, por causa do alto número de veículos sem rota. Seguindo o mapa de ligação, os veículos continuam indo para segmentos que não necessariamente são o melhor caminho para o destino e isso faz com que eles percorram uma extensão bem maior do que a necessária. Nos cenários de caminho único, as distâncias médias são parecidas para todas as versões,

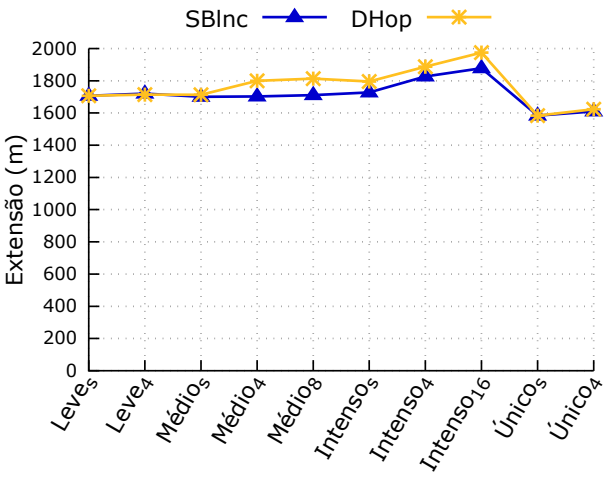
pois, como todos os veículos têm a mesma origem e o mesmo destino, não há como eles se afastarem muito do caminho sem aumentar demais o tempo de viagem.

Figura 29 – Extensão percorrida



Fonte: Resultados da pesquisa.

Figura 30 – Extensão (SBlnc x D-Hop)



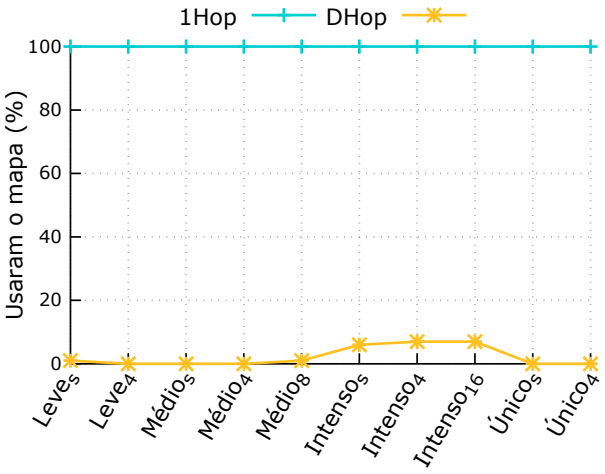
Fonte: Resultados da pesquisa.

Na Figura 30, estão as extensões médias percorridas pelos veículos seguindo a versão sem balanceamento (SBlnc) e o D-Hop. A distância média percorrida pelos veículos seguindo o D-Hop aumentou, principalmente nos cenários em que havia uma necessidade maior de espalhar os veículos para conseguir tempos menores, que são os cenários médio e intenso com acidentes. A distância aumentou com o uso do balanceamento, pois os novos caminhos usados coincidiram de ser maiores que os usados pela versão sem balanceamento.

Na Figura 31, estão as porcentagens do total de veículos que alcançou o destino que precisaram usar o mapa de ligação em algum momento, pelo menos uma vez. Um número maior de veículos seguindo o D-Hop precisou usar o mapa nos cenários intensos em comparação com os outros cenários, pois, como nesses cenários há mais veículos e, portanto, um número maior de mensagens circulando pela rede, nem sempre os veículos conseguiam receber a resposta de rota vinda da RSU e acabavam precisando usar o mapa de ligação. Os veículos seguindo o 1Hop, por outro lado, precisaram todos usar o mapa de ligação. Mesmo pedindo rota a cada 8 segundos, como apenas o próximo salto era enviado por resposta, logo os veículos já precisavam da próxima mensagem e ficavam sem rota por não recebê-la a tempo.

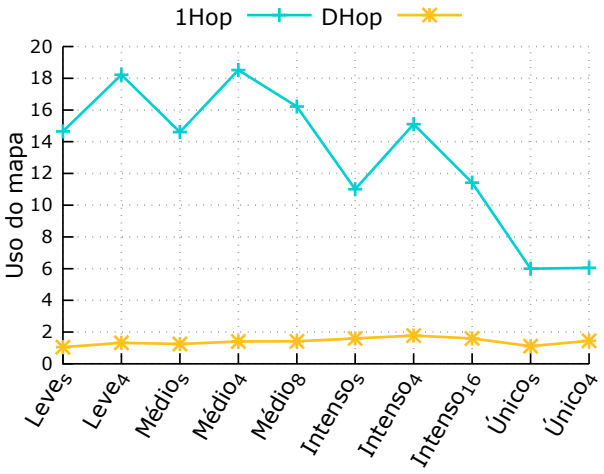
Com relação ao número absoluto de vezes que os veículos precisaram usar o mapa, há uma grande diferença entre o 1Hop e o D-Hop, como ilustrado pela Figura 32. Nos cenários de fluxo leve e médio, os veículos seguindo o 1Hop precisaram usar o mapa até 18 vezes em média, o que explica a extensão consideravelmente maior percorrida pelos veículos desses cenários. Nos cenários de fluxo intenso houve uma diminuição, pois com o aumento do número de veículos eles conseguiam receber a resposta de rota mais frequentemente, já que estavam com velocidades reduzidas (Figura 33) e, portanto, usavam menos vezes o mapa.

Figura 31 – Usaram o mapa (%)



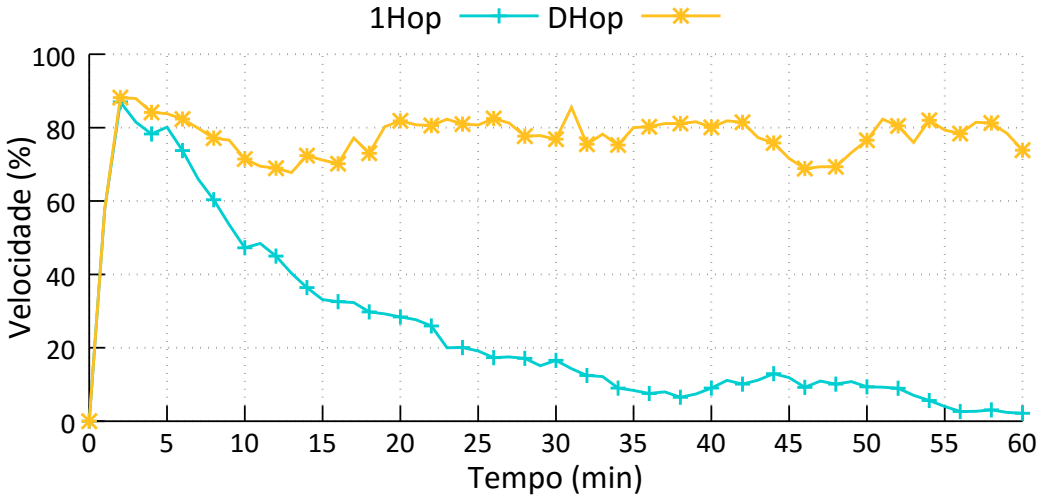
Fonte: Resultados da pesquisa.

Figura 32 – Uso do mapa (absoluto)



Fonte: Resultados da pesquisa.

Figura 33 – Velocidades no tempo do cenário intenso sem acidentes (%)



Fonte: Resultados da pesquisa.

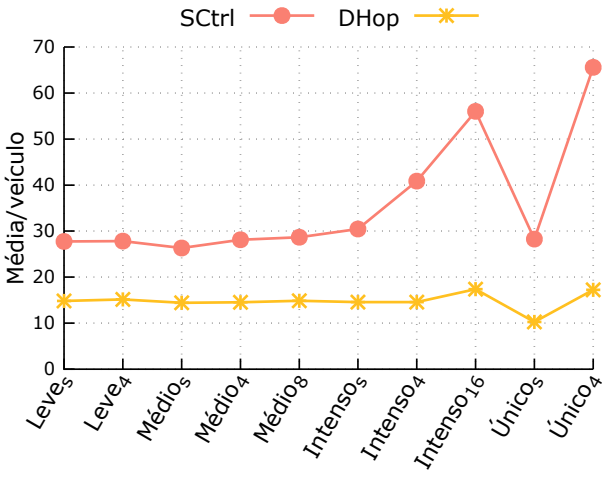
Para avaliar o controle proposto para o pedido de rota, na Figura 34, estão os números médios de mensagens de pedido de rota enviados por veículo que alcançaram o destino e, na Figura 35, estão as mensagens de pedido de rota totais enviadas, também apenas dos veículos que alcançaram o destino. As versões avaliadas nesta métrica foram a sem controle no envio de mensagens (SCtrl) e o D-Hop.

Usando o controle proposto no D-Hop, menos mensagens foram enviadas por veículo do que sem o controle, e houve uma estabilidade entre os cenários, mostrando que o controle se adaptou bem às variações de velocidade de cada cenário. No caso do cenário Único_s, a média foi menor pois os veículos alteram o intervalo de pedido de rota assim que recebem uma resposta da RSU. Como todos os veículos estavam seguindo no caminho único, ou razoavelmente perto dele, e todos queriam o mesmo destino, a resposta por *broadcast* da RSU atendia a esses veículos, que passavam a enviar menos mensagens.

A média dos cenários Intenso_{16} e Único_4 foi maior pois nesses dois cenários nem

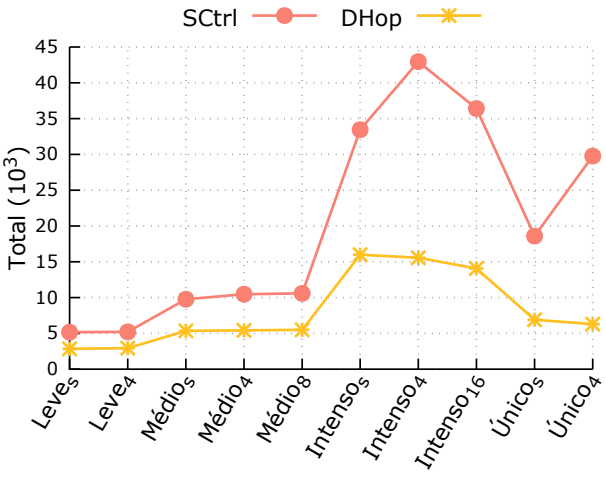
todos os veículos alcançaram o destino, como é possível observar na Figura 36. Isso aumenta a média por veículo, que é o total de mensagens enviadas por todos os veículos que alcançaram o destino dividido pelo número de veículos que alcançou o destino.

Figura 34 – Pedidos de rota por veículo



Fonte: Resultados da pesquisa.

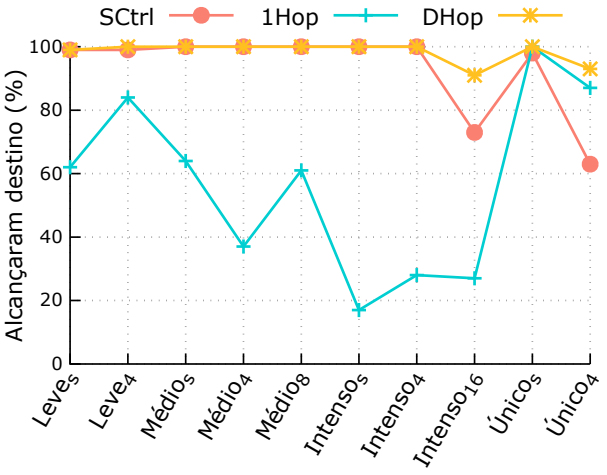
Figura 35 – Total de pedidos de rota



Fonte: Resultados da pesquisa.

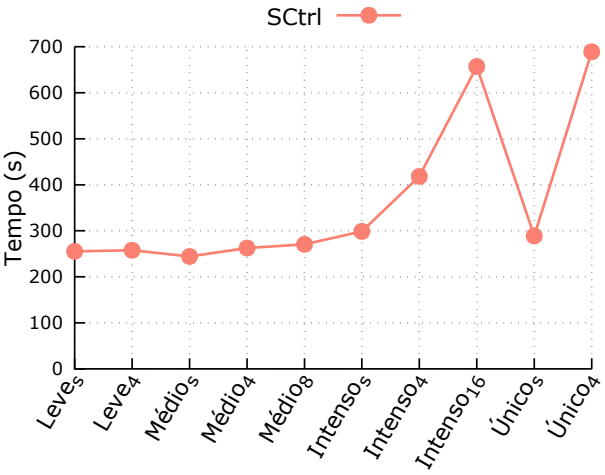
Na versão SCtrl as médias por veículo (Figura 34) nos cenários Intenso_4, Intenso_{16} e Único_4 foram consideravelmene maiores que nos outros cenários, pois, embora o envio das mensagens de pedido de rota sejam constantes nessa versão, o tempo de viagem desses cenários foi consideravelmente maior que dos outros (vide Figura 37), devido aos acidentes simulados. Ainda, no caso dos cenários Intenso_{16} e Único_4, a média também aumentou devido ao menor número de veículos que alcançou o destino (Figura 36). Portanto, embora nesses dois cenários tenham sido enviadas menos mensagens no total que no cenário Intenso_4 (Figura 35), a média deles foi maior (Figura 34).

Figura 36 – Veículos que alcançaram o destino (%)



Fonte: Resultados da pesquisa.

Figura 37 – Tempo de viagem da versão SCtrl



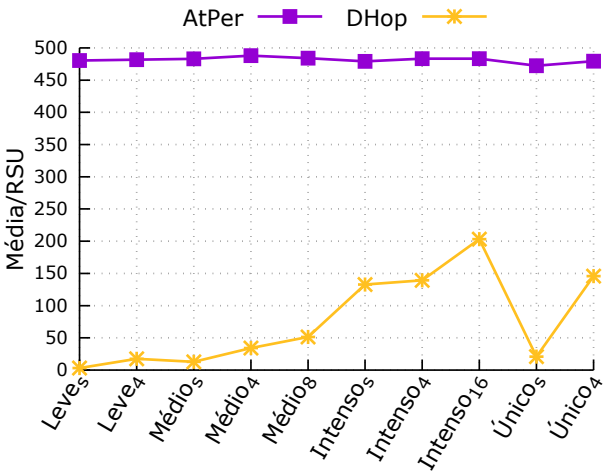
Fonte: Resultados da pesquisa.

Nem todos os veículos conseguiram alcançar o destino durante o tempo de simulação, como ilustrado na Figura 36. Isso está relacionado com o fato de os veículos

não receberem a resposta de rota a tempo das RSUs e com os tempos de viagem. Nos cenários Leve_s e Leve_4, para o D-Hop e a versão SCtrl, o menor número de veículos que alcançou o destino está mais relacionado a eles não receberem rota a tempo, pois, nesses cenários, os veículos passavam mais rápido pelos segmentos e nem sempre conseguiam receber a resposta da RSU.

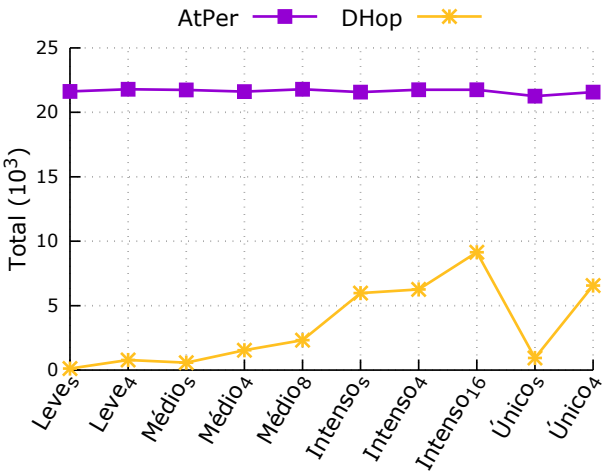
Nos cenários Intenso_{16} e Único_4, também para o D-Hop e a SCtrl, o número menor de veículos que alcançou o destino foi mais influenciado pelo tempo de viagem maior desses cenários (Figuras 24 e 37). Com o aumento do tempo de viagem médio, a simulação acabava antes de esses veículos conseguirem alcançar o destino. Na maioria dos cenários, poucos veículos seguindo o 1Hop chegaram ao destino antes de a simulação acabar, pois passaram a maior parte do tempo sem rota (Figura 31), usando o mapa de ligações e andando a esmo pelo mapa.

Figura 38 – Mensagens de atualização por RSU



Fonte: Resultados da pesquisa.

Figura 39 – Total de mensagens de atualização



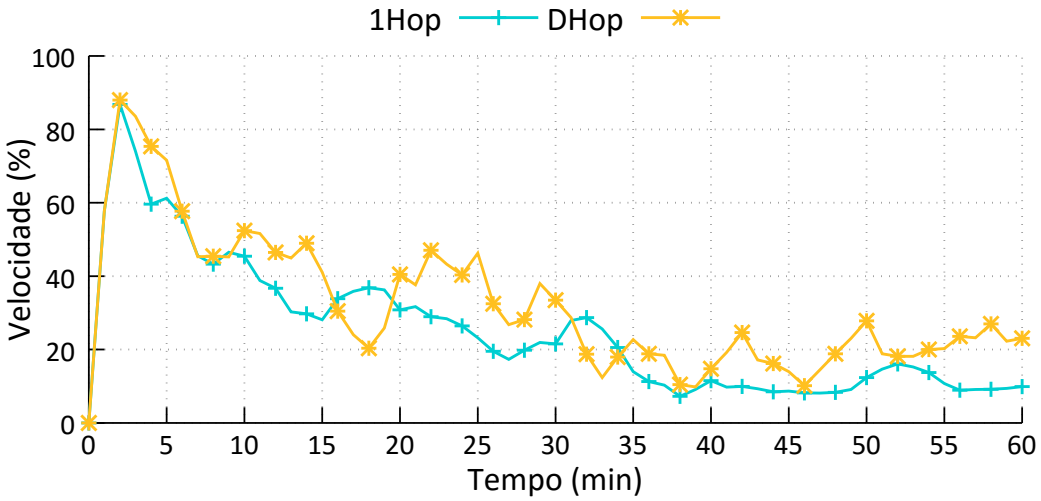
Fonte: Resultados da pesquisa.

Com relação às mensagens de atualização da tabela enviadas, na Figura 38, estão as médias por RSU e, na Figura 39, está o total de mensagens. Como é possível observar, com o controle da atualização da tabela proposto no D-Hop, menos mensagens foram enviadas que do sem o controle (AtPer). Sem o controle, as RSUs enviavam mensagens de atualização da tabela de dois em dois minutos, para todos os segmentos que elas cuidavam, o que causou um número tão maior ser enviado e todos os cenários seguindo a AtPer enviarem quantidades aproximadas.

Usando o D-Hop, o número de mensagens foi aumentando juntamente com a maior variação nas velocidades dos segmentos de cada cenário. Isso é perceptível pela diferença entre os cenários sem acidente e com o número de acidentes aumentando. Na Figura 40, estão as velocidades médias no tempo do cenário Intenso_{16}, para ilustrar a variação intensa entre as faixas de 50 a 20% e 20 a 10% da velocidade. Na Figura 33, por outro lado, é possível observar que a velocidade é constante na faixa de 100 a 50% de velocidade dos

veículos seguindo o D-Hop, o que explica o número menor de mensagens enviadas nesse cenário.

Figura 40 – Velocidades no tempo do cenário intenso com 16 acidentes (%)



Fonte: Resultados da pesquisa.

Quanto maior a variação de velocidade nos segmentos do mapa, mais mensagens as RSUs precisam enviar para manter a tabela atualizada (vide cenário Intenso_{16} na Figura 38). Quando as velocidades estão estáveis, o controle do D-Hop garante que menos mensagens sejam enviadas (cenário Único_{s} na Figura 38, cujas velocidades médias estão na Figura 26).

4.6 Resumo do capítulo

O D-Hop e cinco variações dele foram implementados e avaliados. No geral, o D-Hop apresentou tempos de viagem menores, menor emissão de CO_2, velocidades médias maiores e mais constantes e poucos veículos não receberam rota a tempo das RSUs. O número de mensagens enviadas, tanto de pedido de rota como de atualização da tabela, também foi menor em relação a não usar os controles de mensagem propostos, com reduções médias de 67 e 95%, respectivamente.

Principalmente em comparação ao método da literatura, ICODE, aqui implementado como 1Hop, os ganhos do D-Hop foram: uma diminuição média de 85% no tempo de viagem; 81% de redução da emissão de CO_2, em média; diminuição média de até 54% da extensão percorrida; velocidade média de aproximadamente 80% da máxima contra 10% apresentada pelos veículos do 1Hop; e 100% veículos que alcançaram o destino contra apenas 20% seguindo o 1Hop.

5 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi apresentado o D-Hop, um protocolo dinâmico e distribuído de roteamento de veículos com controle no envio de mensagens. Através do uso de tabelas de roteamento mantidas com a condição atualizada do trânsito, de maneira distribuída, os veículos são conduzidos desde suas origens até seus destinos com tempos de viagem menores e com menos mensagens transmitidas. O D-Hop foi avaliado através de simulações segundo diversas métricas e, em comparação a variações dele próprio e à adaptação (1Hop) do método da literatura citado anteriormente, ICODE, obteve resultados melhores.

Com a distribuição do tráfego criada pelo uso do D-Hop, os veículos simulados conseguiram chegar ao seu destino em menos tempo e com uma velocidade média maior em comparação ao 1Hop. Isso também reduziu a poluição causada pela emissão de CO_2, diminuindo o impacto negativo no meio-ambiente. Além disso, com o D-Hop há menos mensagens transmitidas e menor número de veículos que não recebem sua rota a tempo. A diferença é particularmente notável no tempo de viagem médio dos veículos usando o D-Hop e usando o 1Hop, quando foram obtidas melhoras de, em média, 85% no tempo de viagem dos veículos. O balanceamento proposto no D-Hop, que envia os veículos para segmentos de saída diferentes caso o tempo esteja dentro do limite, apresentou resultados melhores em relação a não usar o balanceamento, com redução de 31%, em média, no tempo de viagem. Os controles de mensagem propostos também cumpriram bem seu papel de conseguir manter o roteamento bom com um número menor de mensagens transmitidas, com reduções de até 95%, em média, do número de mensagens transmitidas.

5.1 Contribuições

O principal diferencial do D-Hop é o controle no envio de mensagens, tanto por parte dos veículos quanto por parte das RSUs. Esse controle funciona através das faixas de velocidade estabelecidas. O número de mensagens de pedido de rota foi diminuído em até 67%, em média, e as mensagens de atualização da tabela em até 95%, em média, tornando a troca de mensagens mais eficaz e eficiente, pois mensagens inúteis não são enviadas. Como, seguindo o D-Hop, as RSUs guardam mais de um caminho possível para cada destino, foi possível implementar o balanceamento de fluxo entre esses caminhos de saída de uma RSU. As RSUs do 1Hop guardavam apenas o caminho de menor tempo, o que tornava esses segmentos sobrecarregados com o tempo, aumentando o tempo de viagem. Com o balanceamento proposto, já foi possível observar reduções no tempo de viagem dos veículos, mesmo que tenha havido um aumento na distância percorrida.

Houve ganhos do D-Hop, também, em relação à atualização da tabela de roteamento. Com a adaptação (1Hop) do método da literatura, ICODE, apenas a entrada

do destino é atualizada com o recebimento de uma nova mensagem. Seguindo o D-Hop, as RSUs atualizam todos os caminhos que passam pelo salto recebido na mensagem, mantendo os tempos de viagem mais atualizados que o 1Hop e, portanto, garantindo o envio dos veículos para os caminhos de menor tempo de fato, pois as RSUs não ficam dependentes de receber uma mensagem de um destino específico para atualizar seu tempo na tabela. O controle de mensagens ajudou a garantir que os veículos não perdessem a resposta da RSU com a rota, pois a rede não ficava sobrecarregada. A grande diferença entre o D-Hop e o 1Hop foi em relação ao número de saltos enviados de cada vez para os veículos. Armazenar nas tabelas e enviar para os veículos cinco saltos de cada vez apresentou resultados excelentes em comparação à adaptação do método da literatura, que guardava apenas um salto. Como os veículos simulados com o D-Hop conheciam mais saltos de cada vez, havia uma necessidade menor de eles receberem uma resposta da RSU em cada segmento, diferentemente dos veículos seguindo o 1Hop.

5.2 Trabalhos futuros

Como trabalhos futuros, é sugerida a comparação do D-Hop com outros métodos de sugestão de rotas, além de propostas que visem melhorar o balanceamento realizado pelas RSUs usando, por exemplo, previsão da densidade ou do tempo de viagem dos segmentos antes de tomar a decisão de enviar veículos para lá. Dessa forma, será possível obter ganhos ainda maiores no tempo de viagem total e impedir a formação de novos congestionamentos, visto que a maior diminuição observada no tempo de viagem com o uso do balanceamento do D-Hop foi de apenas 31%, em média.

Faz-se necessário tornar o D-Hop escalável no número de destinos, pois quanto mais destinos existem, maiores são as tabelas de roteamento. Com muitos destinos possíveis, as tabelas mais distantes de onde está havendo uma situação de trânsito lento atualizam muito devagar, pois as mensagens demoram para chegar em regiões distantes, então as rotas ficam desatualizadas. Para tornar o D-Hop escalável, é possível criar regiões no mapa. Dentro de cada região as tabelas serão mantidas atualizadas sobre a região específica a que elas pertencem, usando o D-Hop como ele é atualmente. Para veículos em uma região que querem chegar a um destino em outra região, as RSUs não saberão o caminho até o destino, mas vão saber conduzir o veículo até uma RSU que saiba.

Deve-se, ainda, testar outros valores para as porcentagens baseadas no número de saltos conhecidos e para as faixas de velocidade definidas para os veículos. Os valores escolhidos para o D-Hop já apresentaram resultados bastante satisfatórios, mas pode-se tentar melhorar o controle de mensagens de pedido de rota e garantir que nenhum veículo fique sem caminho.

REFERÊNCIAS

ALVES, R. d. S. et al. Uma análise experimental da capacidade de redes ad hoc veiculares. In: Simpósio Brasileiro de Telecomunicações (SBrT), 26., **Anais...** Rio de Janeiro: SBrT, 2008. p. 8.

ALVES, R. S. et al. Redes veiculares: Princípios, aplicações e desafios. In: Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 27., **Anais...** Recife: Minicurso do Simpósio Brasileiro de Redes de Computadores, 2009. p. 199–254.

ARAUJO, G. B. **CARTIM:** um protocolo de identificação e minimização de congestionamentos veiculares. Dissertação (Mestrado) — Pontifícia Universidade Católica de Minas Gerais, Programa de Pós-Graduação em Informática, Belo Horizonte, 2014.

BRENNAND, C. A. R. L. et al. An intelligent transportation system for detection and control of congested roads in urban centers. In: IEEE Symposium on Computers and Communication (ISCC), **Proceedings...** Larnaca: IEEE, 2015. p. 663–668.

DENATRAN. **DENATRAN** - frota de veículos no Brasil. 2016. Disponível em: <http://www.denatran.gov.br/frota.htm>. Acesso em: 25 abr. 2013.

DOOLAN, R.; MUNTEAN, G. M. Vanet-enabled eco-friendly road characteristics-aware routing for vehicular traffic. In: IEEE Vehicular Technology Conference (VTC Spring), 77., **Proceedings...** Dresden: IEEE, 2013. p. 1–5. ISSN 1550-2252.

FAHMY, M. F.; RANASINGHE, D. N. Discovering automobile congestion and volume using vanet's. In: International Conference on ITS Telecommunications (ITST), 8., **Proceedings...** Phuket: IEEE, 2008. p. 367 –372.

FIORE, M. et al. Vehicular mobility simulation for vanets. In: Annual Simulation Symposium (AnSS), 40., **Proceedings...** Norfolk: IEEE, 2007. p. 301–309. ISSN 1080-241X.

GTNETS. **The Georgia Tech Network Simulator (GTNetS)** - download. 2008. Disponível em: <http://www2.ece.gatech.edu/research/labs/MANIACS/GTNetS/downloads.html>. Acesso em: 20 jun. 2013.

HARTENSTEIN, H.; LABERTEAUX, K. P. A tutorial survey on vehicular ad hoc networks. **IEEE Communications Magazine**, v. 46, n. 6, p. 164–171, Jun 2008. ISSN 0163-6804.

INSTITUTE, T. A. T. **2015 Urban Mobility Scorecard**. 2015. Disponível em: <http://mobility.tamu.edu/ums>. Acesso em: 15 mar. 2016.

JIST/SWANS. **JiST/SWANS** - Java in Simulation Time/Scalable Wireless Ad hoc Network Simulator. 2004. Disponível em: <http://jist.ece.cornell.edu>. Acesso em: 20 jun. 2013.

LEONTIADIS, I. et al. On the effectiveness of an opportunistic traffic management system for vehicular networks. **IEEE Transactions on Intelligent Transportation Systems**, v. 12, n. 4, p. 1537–1548, Dec 2011. ISSN 1524-9050.

LI, F.; WANG, Y. Routing in vehicular ad hoc networks: A survey. **IEEE Vehicular Technology Magazine**, v. 2, n. 2, p. 12 –22, Jun 2007. ISSN 1556-6072.

LIU, Y.; BI, J.; YANG, J. Research on vehicular ad hoc networks. In: Chinese Control and Decision Conference, 21., **Proceedings...** Guilin: IEEE, 2009. p. 4430–4435. ISBN 978-1-4244-2723-9.

MARTINEZ, F. J. et al. Citymob: A mobility model pattern generator for vanets. In: IEEE International Conference on Communications Workshops, 8., **Proceedings...** Beijing: IEEE, 2008. p. 370–374.

MARTINEZ, F. J. et al. A survey and comparative study of simulators for vehicular ad hoc networks (vanets). **Wireless Communications and Mobile Computing**, v. 11, n. 7, p. 813 – 828, 2011. ISSN 1530-8677.

NAFI, N. et al. A predictive road traffic management system based on vehicular ad-hoc network. In: Australasian Telecommunication Networks and Applications Conference (ATNAC), **Proceedings...** Southbank: IEEE, 2014. p. 135–140.

NS-2. **The Network Simulator 2 (ns-2)**. 2012. Disponível em: <http://www.isi.edu/nsnam/ns>. Acesso em: 20 jun. 2013.

NS-3. **The Network Simulator 2 (ns-3)**. 2014. Disponível em: <http://www.nsnam.org>. Acesso em: 20 jun. 2013.

OMNET++. **OMNeT++** - download. 2016. Disponível em: <https://omnetpp.org/omnetpp>. Acesso em: 20 jun. 2013.

PAN, J. et al. Proactive vehicle re-routing strategies for congestion avoidance. In: IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), 8., **Proceedings...** Hangzhou: IEEE, 2012. p. 265–272.

PAN, J. et al. Proactive vehicular traffic rerouting for lower travel time. **IEEE Transactions on Vehicular Technology**, v. 62, n. 8, p. 3551–3568, Oct 2013. ISSN 0018-9545.

SHAFIEE, K. et al. Modeling and simulation of vehicular networks. In: ACM international symposium on Design and analysis of intelligent vehicular networks and applications, 1., **Proceedings...** New York, NY, USA: ACM, 2011. p. 77–86. ISBN 978-1-4503-0904-2. Disponível em: <http://doi.acm.org/10.1145/2069000.2069014>.

SNS. **A staged network simulator (SNS)**. 2001. Disponível em: <http://www.cs.cornell.edu/people/egs/sns>. Acesso em: 20 jun. 2013.

SOUZA, A. M. de et al. Scorpion: A solution using cooperative rerouting to prevent congestion and improve traffic condition. In: IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), **Proceedings...** Liverpool: IEEE, 2015. p. 497–503.

SOUZA, A. M. de et al. Garuda: A new geographical accident aware solution to reduce urban congestion. In: IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), **Proceedings...** Liverpool: IEEE, 2015. p. 596–602.

STRAW. **STRAW (Street Random Waypoint)**. 2008. Disponível em: <http://www.aqualab.cs.northwestern.edu/projects>. Acesso em: 20 jun. 2013.

SUMO. **SUMO (Simulation of Urban Mobility)** - download. 2016. Disponível em: <http://www.sumo.dlr.de/userdoc/Downloads.html>. Acesso em: 20 jun. 2013.

VANETMOBISIM. **VanetMobiSim:** generating realistic mobility patterns for VANETs. 2006. Disponível em: <http://vanet.eurecom.fr>. Acesso em: 20 jun. 2013.

VEINS. **Veins (Vehicles in Network Simulation)** - tutorial. 2016. Disponível em: <http://veins.car2x.org/tutorial>. Acesso em: 20 jun. 2013.

WANG, M. et al. Real-time path planning based on hybrid-vanet-enhanced transportation system. **IEEE Transactions on Vehicular Technology**, v. 64, n. 5, p. 1664–1678, May 2015. ISSN 0018-9545.

WEDEL, J.; SCHUNEMANN, B.; RADUSCH, I. V2x-based traffic congestion recognition and avoidance. In: International Symposium on Pervasive Systems, Algorithms, and Networks (ISPAN), 10., **Proceedings...** Kaohsiung: IEEE, 2009. p. 637–641.

YOUNES, M.; ALONSO, G.; BOUKERCHE, A. A distributed infrastructure-based congestion avoidance protocol for vehicular ad hoc networks. In: IEEE Global Communications Conference (GLOBECOM), **Proceedings...** Anaheim: IEEE, 2012. p. 73–78. ISSN 1930-529X.

YOUNES, M.; BOUKERCHE, A. A performance evaluation of a context-aware path recommendation protocol for vehicular ad-hoc networks. In: IEEE Global Communications Conference (GLOBECOM), **Proceedings...** Atlanta, GA: IEEE, 2013. p. 516–521.

YOUNES, M.; BOUKERCHE, A. A traffic balanced mechanism for path recommendations in vehicular ad-hoc networks. In: IEEE Global Communications Conference (GLOBECOM), **Proceedings...** Austin: IEEE, 2014. p. 45–50.

YOUNES, M. B. et al. An efficient fault tolerant distributed path recommendation protocol for next generation of vehicular networks. In: IEEE International Conference on Communications (ICC), **Proceedings...** London: IEEE, 2015. p. 5783–5788.

YOUSEFI, S.; MOUSAVI, M.; FATHY, M. Vehicular ad hoc networks (vanets): Challenges and perspectives. In: International Conference on ITS Telecommunications Proceedings, 6., **Proceedings...** Chengdu: IEEE, 2006. p. 761–766.

ZENG, X.; BAGRODIA, R.; GERLA, M. Glomosim: A library for parallel simulation of large-scale wireless networks. **SIGSIM Simul. Dig.**, ACM, New York, NY, USA, v. 28, n. 1, p. 154–161, jul. 1998. ISSN 0163-6103. Disponível em: <http://doi.acm.org/10.1145/278009.278027>.