Universidade Estadual de Campinas
Instituto de Computação

Allan Mariano de Souza

Towards a Personalized Multi-objective Vehicular Traffic Re-routing System

Sistema de Roteamento de Tráfego Veicular Multi-objetivo Personalizado

CAMPINAS
2021

Allan Mariano de Souza

Towards a Personalized Multi-objective Vehicular Traffic Re-routing System

Sistema de Roteamento de Tráfego Veicular Multi-objetivo Personalizado

Tese apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação no âmbito do acordo de Cotutela firmado entre a Unicamp e a University of Bern.

Dissertation presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor in Computer Science under joint-supervision agreement between Unicamp and University of Bern.

Supervisor/Orientador: Prof. Dr. Leandro Aparecido Villas
Co-supervisor/Coorientador: Prof. Dr. Torsten Braun

This copy corresponds to the final version of the thesis presented by Allan Mariano de Souza and supervised by Prof. Dr. Leandro Aparecido Villas and Prof. Torsten Braun.

CAMPINAS
2021

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

Souza, Allan Mariano de, 1992-
So89t Towards a personalized multi-objective vehicular traffic re-routing systems / Allan Mariano de Souza. – Campinas, SP : [s.n.], 2021.

Orientadores: Leandro Aparecido Villas e Torsten Braun.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Computação.
Em cotutela com: University of Bern.

1. Problema de roteamento de veículos. 2. Sistemas inteligentes de veículos rodoviários. 3. Sistemas de comunicação móvel. 4. Sistemas inteligentes de controle. 5. Otimização multiobjetivo. I. Villas, Leandro Aparecido, 1983-. II. Braun, Torsten. III. Universidade Estadual de Campinas. Instituto de Computação. V. Título.

Informações para Biblioteca Digital

Título em outro idioma: Sistema de roteamento de tráfego veicular multi-objetivo personalizado
Palavras-chave em inglês:
Vehicle routing problem
Intelligent transportation systems
Mobile communication systems
Intelligent control systems
Multiobjective optimization
Área de concentração: Ciência da Computação
Titulação: Doutor em Ciência da Computação
Banca examinadora:
Leandro Aparecido Villas [Orientador]
Torsten Braun
Marília Pascoal Curado
Richard Werner Nelem Pazzi
Thiago Henrique Silva
Edmundo Roberto Mauro Madeira
Data de defesa: 04-06-2021
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-5518-8392
- Currículo Lattes do autor: http://lattes.cnpq.br/4615959094966133

Universidade Estadual de Campinas
Instituto de Computação

Allan Mariano de Souza

Towards a Personalized Multi-objective Vehicular Traffic Re-routing System

Sistema de Roteamento de Tráfego Veicular Multi-objetivo Personalizado

Banca Examinadora:

- Prof. Dr. Leandro Aparecido Villas
  Institute of Computing - UNICAMP
- Prof. Dr. Torsten Braun
  University of Bern
- Prof. Dr. Edmundo Roberto Mauro Madeira
  Institute of Computing - UNICAMP
- Prof. Dr. Thiago Henrique Silva
  Universidade Tecnologica Federal do Parana, Brazil
- Prof. Dr. Richard Werner Nelem Pazzi
  Energy Systems and Nuclear Science Research Centre - OntarioTech
- Profa. Dra. Marília Pascoal Curado
  Department of Informatics Engineering - University of Coimbra

The minutes of the defense, which include the signatures of the members of the Examining Committee, are archived by the University of Campinas and the University of Bern

Campinas, 04 de junho de 2021

Resumo

Roteamento de veículos é a chave para fornecer melhor mobilidade veicular. No entanto, considerar apenas as informações de tráfego para recomendar melhores rotas para cada veículo está longe de atingir os requisitos desejados de um bom Sistema de Gestão de Tráfego (TMS), que visa melhorar a mobilidade, a experiência de condução e a segurança de motoristas e passageiros. Neste cenário, abordagens de redirecionamento cientes do contexto e multi-objetivos terão um papel importante na gestão do tráfego, permitindo que os TMSs considerem diferentes aspectos urbanos que podem afetar as decisões de planejamento de rotas, como mobilidade, distância, consumo de combustível, cenário e segurança . Existem pelo menos três questões que precisam ser tratadas para fornecer um TMS eficiente, incluindo: (i) escalabilidade; (ii) eficiência de redirecionamento; e (iii) confiabilidade. Escalabilidade refere-se à capacidade do sistema de entregar o desempenho desejado sem se preocupar com o número de veículos ou o tamanho do cenário. Por outro lado, a eficiência do redirecionamento se refere ao quão bom é o gerenciamento de tráfego da solução. Por fim, a confiabilidade determina o quão confiáveis são as rotas calculadas pelo sistema em relação às mudanças futuras na dinâmica urbana. Desta forma, esta tese contribui com soluções eficientes e confiáveis para atender aos requisitos de futuros TMSs. A primeira contribuição está no desenvolvimento de uma arquitetura escalável para gerenciamento de tráfego baseada em algoritmos distribuídos e cooperativos para detectar o ambiente urbano, estimar aspectos urbanos e redirecionar veículos em tempo real. A segunda contribuição consiste em possibilitar um encaminhamento multi-objetivo eficiente com base nas preferências de cada usuário. Assim, cada usuário pode determinar quais aspectos urbanos serão escolhidos para planejar seu percurso. Ao contrário de outras abordagens de múltiplos objetivos, nossa solução é não determinística, o que diminui a chance de criar pontos de congestionamento adicionais, uma vez que veículos com origem e destino semelhantes potencialmente serão redirecionados por rotas diferentes. A última contribuição desta tese está em melhorar a confiabilidade das rotas calculadas pelos TMSs utilizando um algoritmo de planejamento de rotas que considera as mudanças futuras na dinâmica urbana proposta. A principal vantagem desta solução em relação às soluções da literatura é que o sistema prevê a dinâmica urbana futura (ou seja, mudanças futuras nas condições de tráfego, riscos de segurança, etc.); assim, o sistema sabe de antemão quando algumas mudanças ocorrerão e quanto tempo durarão, computando consequentemente rotas mais confiáveis. As soluções propostas foram amplamente comparadas com outros trabalhos relacionados em diferentes métricas de avaliação de desempenho. Os resultados da avaliação mostram que as soluções propostas são eficientes, escaláveis e econômicas, impulsionando sistemas de gerenciamento de tráfego de última geração.

Abstract

Vehicular traffic re-routing is the key to provide better vehicular mobility. However, considering just traffic-related information to recommend better routes for each vehicle is far from achieving the desired requirements of a good Traffic Management System (TMS), which intends to improve mobility, driving experience, and safety of drivers and passengers. In this scenario, context-aware and multi-objective re-routing approaches will play an important role in traffic management, considering different urban aspects that might affect path planning decisions such as mobility, distance, fuel consumption, scenery, and safety. There are at least three issues that need to be handled to provide an efficient TMS, including: *(i)* scalability; *(ii)* re-routing efficiency; and *(iii)* reliability. *Scalability* refers to the ability of the system to deliver the desired performance without carrying about the vehicles' number or the scenario's size. On the other hand, *re-routing efficiency* refers to how good is the traffic management of the solution. Finally, *reliability* determines how reliable the system computes the routes regarding future changes in the urban dynamics.

In this way, this thesis contributes to efficient and reliable solutions to meet future TMSs. The first contribution lies in developing a scalable architecture for traffic management based on distributed and cooperative algorithms for sensing the urban environment, estimating urban aspects, and re-routing vehicles in real-time. The second contribution relies on enabling an efficient multi-objective re-routing based on each user's preferences. Thus, each user can determine which urban aspects will be chosen to plan its route. Unlike other multi-objective approaches, our solution is non-deterministic, which decreases the chance of creating additional congestion spots since vehicles with similar origin and destination potentially will be re-routed through different routes. This thesis's last contribution lies in improving the reliability of the routes computed by the TMSs using a route planning algorithm that considers the future changes in the urban dynamics is proposed. The significant advantage of this solution regarding literature solutions is that the system predicts future urban dynamics (i.e., future changes in traffic conditions, safety risks, etc.). Thus, the system knows beforehand when some changes will happen and how long they will last, consequently computing more reliable routes.

The proposed solutions were widely compared with other related works on different performance evaluation metrics. The evaluation results show that the proposed solutions are efficient, scalable, and cost-effective, pushing forward state-of-the-art traffic management systems.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Urban mobility became an evident problem in large cities around the world due to the high number of vehicles on the roads and also by the traffic jams produced by them [46]. One effective way to deal with such problems is the implementation of vehicular traffic re-routing services [62], which aim to improve the overall traffic efficiency by recommending faster routes (e.g., paths that avoid traffic jams) to the vehicles. However, considering traffic-related information to recommend better routes for each vehicle is far from achieving the desired requirements of future Traffic Management Systems (TMS). In this sense, context-awareness and multi-objective re-routing will play an essential role in vehicular traffic re-routing, improving the vehicular experience and enabling a whole new set of services. Hence, improving traffic mobility and driving experience, the energy consumption of electric vehicles, as well as the safety of drivers and passengers [31, 27, 28]. This new vehicular experience will ultimately have a profound impact on society and the daily lives of billions of people worldwide, changing the way we live, work, and play.

1.1 Motivation

Context-awareness in vehicular re-routing is essential since different drivers can have different preferences to perform their journey [4]. These preferences are related to urban factors, including travel time, distance, fuel consumption, scenery, and even safety, which can lead to different routes to reach the same location [4, 62]. However, considering a single preference to re-route vehicles can directly lead to other significant concerns. For instance, [12] shows a real example, where a couple got shot, and the woman died after taking the directions recommended by a vehicular navigation system (VNS), which guided them towards a dangerous neighborhood in Rio de Janeiro, Brazil, aimed to provide the fastest route. Another example shows a vehicle (which took the directions recommended by a VNS) passing through shooting in Boston [52]. These issues could have been possibly avoided using safe route recommendation systems [60, 34]. Navigation systems that focus on optimal safety can lead to stressful paths since the recommendation algorithm can include congested roads to provide the safest way, and these roads are more likely to be avoided in drivers' criteria during their route planning [62]. In this scenario, multi-objective optimization of traffic efficiency and safety is desirable to increase the appeal

and the effectiveness of the re-routing strategy.

Advances in wireless communication and processing such as the fifth-generation (5G) networks [32], vehicle-to-everything (V2X) communication [11] and multi-access edge computing (MEC) [46] will enable TMSs to sense and act in the urban environment in different ways, interacting not only with vehicles, but also with intelligent devices, subsystems, and even people in order to provide better solutions [40]. In other words, TMSs will understand a set of different urban factors and build various pieces of knowledge to help traffic management decisions, including detecting areas with recurrent traffic congestion and limited safety, improve re-routing effectiveness, and avoid dangerous neighborhoods. Besides, with the help of machine learning techniques [67], TMSs can predict future urban dynamics and know in advance when some areas can become congested or dangerous to improve their effectiveness. However, how to predict these urban dynamics accurately and explore their spatiotemporal correlation is still an open issue. In this scenario, deep learning techniques such as recurrent neural networks (RNN) [43, 47, 57, 51] can play an essential role by providing accurate predictions about urban dynamics such as traffic conditions and safety risks while exploring their spatial and temporal information.

Several solutions have been proposed to enable context-awareness and multi-objective re-routing [50] such as Weighted-Sum [49], Resource-Constrained Shortest Path (RCSP) [39] and Evolutionary algorithms [44]. However, these solutions are deterministic and are not suitable for traffic management applications since many vehicles with the same origin and destination can take the same route, potentially degrading traffic efficiency [16]. Besides, most of the TMSs proposed to perform vehicular traffic re-routing have issues related to scalability as a result of the network overhead produced by them [33, 19, 9], and also due to computing efforts related to the complexity time of the re-routing algorithm and their architecture [9, 22, 29, 66, 64].

In this scenario, the major limitations presented by literature solutions to enable multi-objective vehicular traffic re-routing can be listed as follows:

- **High network overhead**: To enable efficient vehicular traffic re-routing, the TMS needs to have accurate knowledge about traffic conditions on the roads. In this way, vehicles need to report their position and velocity to the system periodically to enable traffic condition estimation. However, if all cars communicate their traffic-related information, it potentially overloads the network in dense scenarios, consequently introducing an undesired latency, which might degrade the overall system performance.

- **High computational efforts**: The vehicular re-routing task of a TMS consists of finding better paths connecting an origin and destination pair. Therefore, the complexity time of a personalized re-routing algorithm is directly related to its preferences (e.g., the number of additional urban factors that will be considered to plan the path) and the number of vehicles that need to be re-routed. Consequently, depending on the TMS architecture, such complexity can dramatically increase, especially under heavily congested scenarios.

- **Lack of knowledge about future urban dynamics and their spatiotemporal**

correlation: Different urban factors might have different conditions depending on the region, day, and time, which means that the same area can provide a set of different situations throughout the day for each urban factor (e.g., spatiotemporal correlation). For instance, some areas are more likely to present traffic congestion during rush hours than on business days. Also, some regions can provide different opportunities for criminal activities along the day, either increasing or decreasing the safety risk in that area. Therefore, the lack of knowledge about such a correlation can directly reduce the effectiveness and the reliability of the re-routing algorithm due to dynamic changes.

- **Non-adaptable multi-objective re-routing**: The multi-objective re-routing algorithm itself is not enough to enable personalized re-routing since the TMS needs to be able to understand the relative preferences of each driver to provide methods to allow efficient and customized re-routing. For instance, considering safety risk as a preference, there are a set of events (e.g., criminal actives) that can provide different threats to the safety of drivers and passengers according to their relative preferences. In other words, different drivers can also have different relative choices to each type of crime (e.g., narcotics, assault, shooting, kidnapping, sexual attack, etc.) that can increase or decrease the risk to them. Those relative preferences can also vary according to age, gender, and social characteristics. Thus, enabling each driver its relative choices is essential to achieve trust and reliable TMS.

1.2 Goals and Contributions

Motivated by the limitations of literature solutions and the high safety risks to drivers and passengers produced by public safety issues, this thesis proposes a context-aware vehicular re-routing system to improve traffic efficiency and the safety of drivers and passengers. The system implements mechanisms to extract knowledge about traffic conditions and public safety issues. Thus, personalized path planning is applied, enabling the drivers themselves to choose which safety risk (e.g., criminal events) they want to avoid. Besides, to facilitate decision-making (e.g., compute an alternative route) in advance, the system exploits the spatiotemporal information of each criminal activity, consequently understanding the future safety dynamics of each area. In this context, the goals of this thesis can be achieved by answering the following research questions:

1. **How to ensure system scalability with low overhead and reasonable traffic management?**

   To enable a scalable system and efficient traffic management a distributed traffic-aware data sharing protocol is proposed, in which the vehicles estimate the traffic conditions on the roads locally within their communication range based on the traffic information shared by them. Thus, the best vehicles are chosen to report their local estimations to the TMS by employing a selection mechanism. To enable real-time traffic management (i.e, re-routing) an offloading mechanism is employed to distribute the re-routing task in several processing units. Therefore, by offloading

the re-routing computation in each vehicle, the TMS will overcome the limitation of high latency produced by computational efforts related to the vehicles' density, consequently enabling real-time re-routing and improving system scalability. Finally, a cooperative re-routing approach is proposed to allow that the distributed approach achieves similar performance to the centralized one in terms of traffic balancing effectiveness. The detailed description of these solutions are presented in Chapter 4

2. **How to enable an efficient and personalized multi-objective re-routing without creating different congestion spots?**

   To allow a personalized multi-objective re-routing, an architecture was designed that enables the estimation of different urban dynamics, such as, travel time, green house emissions, and safety. Then, by extracting knowledge about city dynamics, the system can implement multi-objective re-routing. For instance, by extracting information about traffic conditions and safety risks, the system re-routes vehicles through the fastest and safest route. Also, supported by the offloading mechanism to enable real-time re-routing, vehicles can decide which information they want to re-route themselves. Therefore, the system provides the vehicles the info they are interested in. However, to achieve the desired effect of traffic re-routing and avoid creating different congestion spots (due to many vehicles with the exact origin and destination take the same route). A non-deterministic re-routing algorithm is proposed to compute the set of paths that improve mobility of the vehicle and safety considering its preferences and then distribute the traffic flow over the collection of routes previously computed. The detailed description of these solutions to deal with personalized multi-objective re-routing for TMS is presented in Chapter 5

3. **How to consider future changes in urban dynamics during re-routing to plan more efficient and reliable routes?**

   To pave the way for a more efficient and reliable traffic re-routing a prediction model is proposed, which predicts future changes in the urban dynamics to understand the environment and to know when and where some changes will happen. Thus, a recurrent neural network has been implemented to predict the future dynamics (considering a time window) based on past events and historical data. Hereafter, a routing algorithm that considers the predicted dynamics during the route planning is proposed. In summary, the algorithm learns the most efficient path by iterating with the environment considering the future changes using a trial-and-error approach, which is suitable for reinforcement learning-based solutions. In other words, a reinforcement learning-based route planning algorithm is proposed. The detailed description of the prediction model and the reinforcement learning-based algorithm are presented in Chapter 6

It is worth noticing that the solutions presented in this thesis are published in relevant conferences and prestigious journals on computer networks and communication. The conferences in which the solutions were published include *IEEE International Conference on Communications (ICC), ACM International Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWIM), IEEE Vehicular Technology Conference*

(VTC), IEEE Intelligent Transportation System Conference (ITSC), IEEE Distributed Computing in Sensor Systems (DCOSS), and Brazilian Symposium on Computer Networks and Distributed Systems (SBRC). On the other hand, the published journals: *IEEE Transactions on Intelligent Transportation Systems, IEEE Intelligent Transportation System Magazine, Elsevier Ad hoc Networks, Springer Journal of Internet Applications and Services, MDPI Sensors.* The complete list of publications and papers under review are presented in Chapter 7 in Section 7.3.

1.3 Thesis Structure

The thesis is organized as follows. Chapter 2 provides the background for this thesis describing how the communication technologies can improve the TMS performance and how machine learning techniques can provide better services for TMS. Chapter 3 describes the related work highlighting the limitations and advantages of literature solutions for traffic management. Moreover, qualitative comparison and classification are proposed. Chapter 4 presents an efficient and scalable, and cooperative solution for vehicular traffic re-routing, which aims to answer the first research question presented in the previous section. Chapter 5 introduces the solution for personalized multi-objective re-routing using a non-deterministic algorithm, which considers the spatiotemporal correlation about urban dynamics. This aims to answer the second research question presented in the previous section. Chapter 6 presents an efficient solution for reliable route planning that considers future changes in urban dynamics to plan more efficient routes. This solution aims to answer the third research question presented in the previous section. At last, Chapter 7 presents conclusions, describes the future work, and describes potential research opportunities provided by the studies presented in this thesis.

Chapter 2

Background

2.1 Traffic Management Systems

The ultimate goal of TMS is to understand urban conditions in real-time and derive solutions to improve the driving experience, mobility, and safety [46]. To do so, TMSs can rely on the plethora of sensors and smart objects that will be present in the smart cities and emerging communication and information technologies. Figure 2.1 shows the future landscape of TMSs, presenting how sensing, communication, and information technologies can work together to improve the efficiency and reliability of services provided by a TMS. Specifically, these technologies can help to overcome the major issues related to TMS such as: *(i)* real-time urban sensing; *(ii)* low-latency communications and massive data storage; and *(iii)* future urban dynamic predictions and real-time responsiveness.

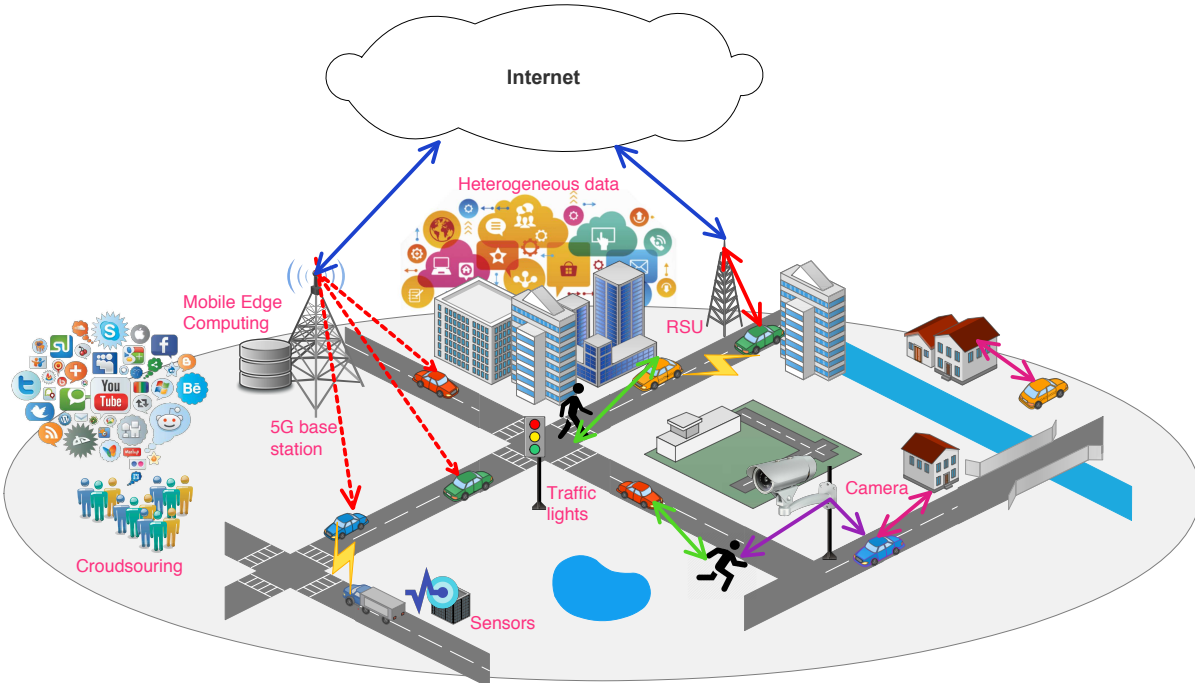


Figure 2.1: TMS landscape

Real-time sensing: Urban sensing plays a vital role in TMS since it is the foun-

dation of the services that will be delivered [46]. Currently, environment sensing relies on measuring traffic conditions based on data gathered by sensors deployed at vehicles and roadside infrastructures [31]. However, the massively connected world enabled by the Internet of Things (IoT) and smart-objects will allow a new sensing paradigm. The TMSs can have a deep understanding of different urban factors, ranging from physical to social features and behaviors. Such knowledge will be achieved by exploiting the considerable volume of heterogeneous data produced by sensors, smartphones, wearable devices, smart objects, sub-systems, and even people present in the urban environment.

Low-latency communications and massive data storage: The massive amount of raw data produced by sensors can easily touch the petabyte order size [31]. Given the difference in data types, dimensionality, and huge volume, the bandwidth of communication networks, storage capability, and data processing speed need to be as fast and efficient as possible to avoid decreasing the TMS performance. In this scenario, vehicle-to-everything (V2X) communications [11], fifth-generation 5G networks (5G) [32], Visible Light Communication (VLC) [13], and Multi-access Edge Computing (MEC) [46] will be essential to achieve the desired performance of a TMS with its strict requirements.

Urban dynamics prediction and real-time responsiveness: Unexpected incidents and sudden changes in urban dynamics can affect TMS services. For instance, traffic accidents might degrade overall traffic efficiency and also lead to personal damages and even deaths, which are grave concerns [31]. Thus, providing methods for handling such problems at an appropriate time to avoid further damages is crucial for TMS. Moreover, being able to handle some specific problems beforehand, such as traffic congestion and public safety issues, can significantly improve the efficiency of TMS. Therefore, machine learning techniques will be essential to predict future urban dynamics and provide helpful information to TMSs in advance. In other words, machine learning techniques can provide proactive management into such a reactive environment.

Table 2.1 summarizes how the emerging communication information technologies can address the major issues related to TMS.

Table 2.1: Summary of information and communication technologies to enable reliable TMS.

Area	Technology	Enabling role
Sensing	**In-vehicle sensing**: OBD-II, Smartphones **Off-vehicle sensing**: road sensors, roadside infrastructures **Extra-sensing sources**: weather reports; crowdsourcing; police/incident reports	Allows for enhanced understanding of urban context Facilitates Urban dynamics and driver behavior modeling
Communication	Wifi, DSRC, 802.11p, 5G, VLC	Allows for inter-vehicle information exchange Allows cooperative tasks Enables offloading computing processes from/to vehicles Enables high bandwidth and low-latency communications
Processing	Tiered in-network computing Multi-access edge computing Cloud computing	Facilitates robust and comprehensive TMS solutions Enables an advantageous computing hierarchy Supports efficient decision making Expand storage and computing capabilities of TMS

2.2 Communication and Processing Technologies for TMS

This section briefly describes the emerging communication and processing technologies that will empower TMS to improve its services. These technologies include vehicle-to-

everything, 5G networks, visible light communications, and multi-access edge computing.

2.2.1 Vehicle-to-Everything

V2X defines four types of communication modes such as vehicle-to-vehicle (V2V), vehicle-to-pedestrian (V2P), vehicle-to-infrastructure (V2I) [11].

V2V and V2P modes cover direct communication among vehicular user equipment, vehicles, and vulnerable road users, such as pedestrians, bikers, motorcyclists, and wheelchair users. In V2V communications, vehicles can use a set of different technologies to establish direct links, including WiFi, Dedicated Short Range Communication (DSRC), IEEE 802.11p, and VLC [5, 32, 11]. On the other hand, considering V2P, vehicles can use WiFi, Bluetooth, and other nontrivial communication ways such as through Lidar and cameras, which enable pedestrians to communicate with cars through gestures [45].

V2I refers to communications between vehicles and the roadside infrastructure, for example, a roadside unit (RSU) implemented in an eNodeB, IEEE 802.11p infrastructure, cellular towers, 5G base stations, and access points. Therefore, vehicles and these infrastructures can exchange data over different communication interfaces. Also, these infrastructures can transmit data toward multiples vehicles through broadcast messages such as the evolved multimedia broadcast multicast service (eMBMS) [70]. Last, V2N puts cars in communication with a server supporting TMS applications, referred to as application server, which provides centralized control and the distribution of traffic, road, and service information. The TMS application can be placed either into edge servers or into some remote cloud [46].

2.2.2 Visible Light Communication

VLC uses visible light (380 – 780 nm) as a carrier for the data, and thus it offers a 1000 times greater bandwidth compared to radiofrequency (RF) communications. The visible light spectrum is not regulated, and, therefore, the cost of the technology is significantly reduced. The vast available spectrum enables VLC to achieve very high data rates that can currently go up to few tens of Gb/s [38]. Moreover, since this data rate has been made in less than a decade after the beginning of VLC systems development, it is evident that the technology's potential is even more significant. Furthermore, higher data rates could be achieved by using multiple-input multiple-output (MIMO) communication techniques. These characteristics offer VLC to be part of future 5G technologies [5].

A significant difference between VLC and RF communications comes from the inherent properties of electromagnetic waves. The RF waves can penetrate through most nonmetallic materials, whereas visible light can only penetrate transparent materials. Even though in some cases, the limited penetration capability acts as a disadvantage by limiting the mobility or the coverage area, it could also represent a significant benefit since it limits the interferences between the non-Line-of-Sight (nLoS) systems and prevents eavesdropping.

In addition to the benefits mentioned above, one of the most significant advantages of VLC is the ubiquitous character. In VLC, the data transmission capacity is enabled by fast switching light-emitting diodes (LEDs) as an additional function besides the light-

ing. Thus, the data is transmitted onto the instantaneous power of the light at speeds unperceivable by the human eye. Since it is mainly based on the already existing lighting infrastructure, VLC has the potential to provide high-speed wireless communications wherever there is artificial lighting, indoor and outdoor [13].

2.2.3 5G Networks

5G will be a promising technological breakthrough for the development of vehicular networks. Concerning 4G, the most superior communication technology deployed on a large scale worldwide, 5G is expected to achieve much better performance, including 1000 times higher system capacity, 10–100 times higher number of connecting devices, and user data rate, ten times longer battery life, and five times lower latency [32]. Besides, technologies such as MIMO, cognitive radio (CR), millimeter-wave (mmWave), and heterogeneous networks (HetNets) can work together to address the challenging requirements in vehicular networks.

5G communication technology should be a solid enabler to empower the real-time services of vehicular communications. However, no single technology of 5G can flexibly accommodate the broad range of requirements of vehicular networks. For example, mmWave links will address the obstacle of line-of-sight (LOS) connection, while D2D communications can help maintain links between communicating vehicles/devices. Recent 5G enabled vehicular networks research has demonstrated that a multi-tier and HetNet architecture with the aggregation of various communication technologies could help to achieve the ambitious goals of 5G vehicular networks, and consequently TMSs [70].

2.2.4 Multi-access Edge Computing

The concept of MEC is defined as a new platform that provides cloud-computing capabilities within the Radio Access Network (RAN) near mobile subscribers [48, 46]. The original definition of MEC refers to the use of base stations for offloading computation tasks from mobile devices. However, in future smart cities, ultra-dense edge devices, including small-cell base stations, wireless access points, laptops, tablets, and smartphones, intelligent vehicles will be deployed, each having a considerable processing power [48]. Besides, many of these edge devices will idle instantly, consequently enabling them to exploit their processing (e.g., offload processing tasks to end devices) and storage capabilities available at the network edges, which will be sufficient to allow ubiquitous edge computing. The main features of MEC that will enhance TMS services include:

- **Proximity**: being deployed at the nearest location to vehicles, MEC has the advantage to analyze and materialize big data analysis, which will be beneficial for processing intense services, such as augmented reality, video analytic, and deep learning.

- **Low-latency**: MEC services are deployed at the nearest location to vehicles, isolating network data movement from the core network. This way, TMS can provide high-quality services with ultra-low latency and high bandwidth.

- **Context awareness**: MEC servers leverage the proximity of edge devices to end-users to track their real-time information such as behaviors, locations, and environments. Thus, the inference based on such information allows the delivery of context-aware services to vehicles.

2.3 Machine Learning for TMS

Using machine learning TMS will perform intelligent operations without being explicitly programmed. This will be possible by using algorithms to learn from data and make data-driven decisions or predictions. These approaches can be broadly classified into three main categories: supervised, unsupervised, and reinforcement learning. Some variants of these approaches include semi-supervised, online, and transfer learning. Table 2.2 presents an overview of machine learning approaches with applications in TMS.

In summary, machine learning approaches are composed of two phases, each one working as follows:

- **Training**: derives a model based on the data provided;
- **Predicting**: applies the model to produce predictions.

Table 2.2: Overview of machine learning approaches and applications for TMS.

	Goal	**Algorithms**	**Applications**
Supervised learning	Classification	KNN, Neural Networks, SVM, Decision tree	Traffic flow quality prediction, congestion level
	Regression	Logistic regression, SVR, Gaussian implementations	Urban environment dynamics prediction traffic/Weather conditions/Safety risks
Unsupervised learning	Clustering	K-means, DBSCAN, GMM, Hierarchical clustering	Abnormal driving behavior detection, Anomalies and failure detection in vehicular components
	Dimension reduction	Linear and Nonlinear projection, PCA, Isometric mapping	Data aggregation, traffic condition representation
Reinforcement learning	Policy learning	Q-learning	Self-driving vehicles, Traffic light phase detection, Collision avoidance, Object and person identification, Path planning considering temporal variations

2.3.1 Supervised Learning

Supervised Learning aims to understand how to map a set of input features to its corresponding output. Thus, the data set provided for this approach must be labeled. Each label is either a discrete or continuous value representing the correct prediction for its related input. Discrete values are used by classification algorithms, while regression algorithms use continuous ones.

Classification algorithms rely on the assignment of a class label (category) based on its input samples. On the other hand, regression algorithms forecast the continuous value of some input sample. In TMS, classification algorithms are used to classify urban dynamics such as traffic conditions and congestion levels and also network conditions. Some classification algorithms include Bayesian classifiers, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), decision trees, and Neural Networks (NN) [67].

Regression algorithms for TMS aim to predict the behavior of several issues that can potentially conflict with vehicle trajectory choices, such as traffic dynamics, weather conditions, and even safety risks. Also, they can be used to predict channel parameters and network throughput of available communication technologies, enabling better service performance. Regression algorithms include logistic regression, Support Vector Regression (SVR), and some Gaussian implementations [67].

2.3.2 Unsupervised Learning

Unsupervised Learning focuses on finding an efficient and meaningful representation of the data samples based on their structures, features, and correlations without knowing what each data sample represents. In other words, these approaches aim at learning models with unlabeled data samples.

Clustering is a well-known unsupervised learning approach, which aims at joining up samples into different groups (clusters) based on their similarities. The input features of each sample can be either their absolute description or relative similarities between them. In TMS, clustering algorithms can identify abnormal driving behavior such as wrong directions, aggressive driving, and control loss. Also, they can be used to detect anomalies and failures in-vehicle sensors and internal components. Classic algorithms include k-means, Density-Based Spatial Clustering of Applications with Noise (DBSCAN), Gaussian Mixture Models (GMM), and hierarchical clustering [67].

Dimension reduction is another unsupervised approach that transforms samples with several dimensions into smaller ones by changing their projecting space without losing their representation. Reducing the number of dimensions is important because the algorithms' complexity is directly related to the dimension space. Thus, high dimensional data dramatically increases the algorithms' learning performance. Another reason is that many dimensions are related to each other. Some might have errors or noise, potentially degrade the learning performance if not appropriately addressed.

In TMS, dimension reduction algorithms can be applied to aggregate the vehicles' data and the sensing devices to reduce the overhead and communication cost. Also, in traffic flow prediction, a massive amount of heterogeneous traffic-related, which comes from different sources (vehicles, sensors, smartphones, social media, and people), may represent the same traffic condition; thus, they could be summarized enable more efficient learning algorithms. Some dimension reduction algorithms include linear and nonlinear projection methods and principal component analysis (PCA), and isometric mapping [67].

2.3.3 Reinforcement Learning

Reinforcement Learning learns how to behave in specific situations through a set of trial-and-error interactions in a dynamic environment. Each interaction performs an action on the environment given a situation which results in a reward. Thus, the primary goal is to maximize the expected reward based on a set of actions assuming a Markov Decision Process (MDP).

Q-learning function is used to solve MDP problems, consisting of a model-free learning approach that does not need to know anything about the environment. The Q-learning function estimates the expectation of the resulting reward when taking a set of actions in a given situation (state). The optimal reward is obtained as a maximum expected reward reachable by performing actions regarding a choosing policy.

This approach is applied in TMS to enable self-driving cars, which can learn how to drive with the help of environment sensing information [3, 45]. Thus, self-driving cars can drive without any human intervention, divert objects and people, detect traffic light phases, and avoid collisions. However, it can also be used in path planning and navigation services to perform efficient decisions considering temporal variations in urban dynamics.

2.3.4 Deep Learning

Deep Learning paved the way for significant advances in several machine learning tasks. It is a deep version of neural networks composed of multiple layers, each one comprised of a non-linear information processing unit [47]. Deep learning focuses on understanding the data representations and can be applied to all learning approaches (supervised, unsupervised, and reinforcement), helping a designer establish mapping functions for operation convenience.

Thanks to high-performance computing and faster resources, deeper architectures are becoming feasible, enabling new training methods and structures to greatly improve state-of-the-art approaches in several areas, including computer vision, speech recognition, and natural language processing [47]. Also, depending on the application, different structures can be added to deep networks to improve their performance, such as convolutional neural networks (CNNs) for exploring spatial correlations and recurrent neural networks (RNNs) and long short-term memory for exploring long temporal correlations [47].

For TMS, CNNs have been used to improve image processing tasks such as road surface, object detection, and traffic estimations. On the other hand, RNNs can be applied to improve traffic flow prediction and prediction of urban dynamics by considering the interrelation among temporal variations.

RNN is a deep learning approach that extends the traditional feed-forward networks with internal cycles [43]. These internal cycles support recurrent neural networks with an internal state capable of tracking sequences of information and learning relevant temporal features by exploring intercorrelation among current and past inputs. The LSTM neural network architecture is widely used due to its resistance to the vanishing gradient problem and ability to handle long-term dependencies providing more accurate predictions [51].

The LSTM is composed of one input layer, one recurrent layer whose primary is a memory block, and one output layer. The memory block contains memory cells with

self-connections to memorize the temporal state and three adaptive, multiplicative gating units: the input, the output, and the forget gates to control the information flow in the memory. These gates learn how to open and close, consequently enabling the LSTM cells to store and access information over long periods. Figure 2.2 gives an illustration of the LSTM memory block.

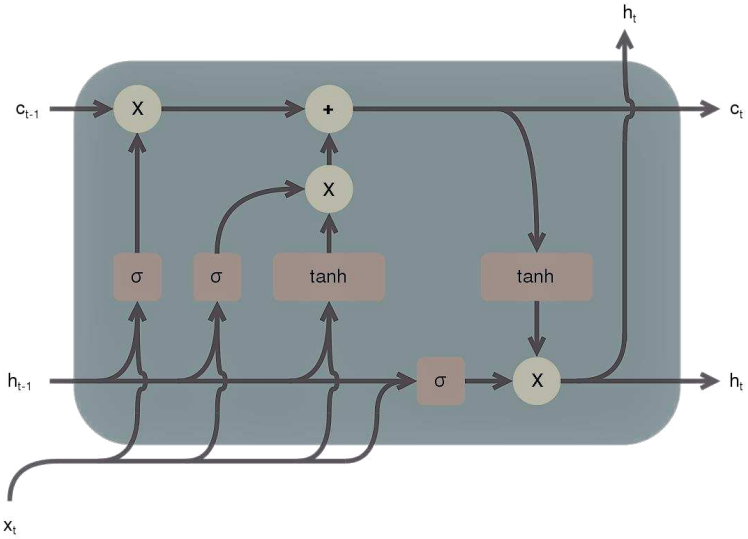


Figure 2.2: LSTM architecture.

Let $x = \{x_1, x_2, x_3, \ldots, x_t\}$ be a sequence representing some time series historical data, and $c = \{c_1, c_2, c_3, \ldots, c_t\}$ represent the LSTM memory block, where T represents the prediction period. Therefore, the prediction of a time series will be represented by h_t, and it can be computed iterative throughout the following equations:

$$h_t = W_{ch}c_t + b_h \tag{2.1}$$

$$c_t = C(W_{x_c} + W_{c_c}c_{t-1} + b_c), \tag{2.2}$$

where W denotes the weight matrices (e.g., W_{x_c} is the input hidden weight matrix), b denotes the bias vectors, and C is the hidden layer function, which can be computed in the following equations:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2.3}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2.4}$$

$$c_t = f_t c_{t-1} + i_t g(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.5}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{2.6}$$

$$h_t = o_t h(c_t) \tag{2.7}$$

i, f, o, and c are the input gate, forget gate, output gate and activation vectors respectively. Where, $\sigma(\cdot)$ is the standard logistic sigmoid function defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.8}$$

In this way, the objective function is to minimize the sum of the square errors, which is given by the following loss function:

$$e_t = \sum_{t=1}^{n} (h_t - y_t)^2, \tag{2.9}$$

where y_t and h_t represent the ground truth of and the predicted value of a specific urban dynamic at time t.

Chapter 3

Related Work

This chapter presents a classification, review, and qualitative analysis of literature TMSs for traffic re-routing. The classification was proposed is based on the re-routing design of each solution. Thus, the solutions were broadly classified in two groups: *single objective re-routing*, solutions in which the re-routing is performed based on only one urban aspect (i.e., travel time, safety risk, fuel consumption, CO_2 emission, etc.); and *(i) multi-objective re-routing*, which are solutions that consider more than one urban aspect to re-routing vehicles. Figure 3.1 shows the proposed classification based on key characteristics of each solution. It worth noticing that the rectangles in green with dashed borders represent the solutions developed by this thesis.

In this way, this chapter is organized as follows. Section 3.1 describes the solutions based on single objective re-routing. Section 3.2 describes the solutions based on multi-objective re-routing. Section 3.3 presents a qualitative analysis of the literature solutions for traffic. At last, Section 3.4 concludes the chapter.

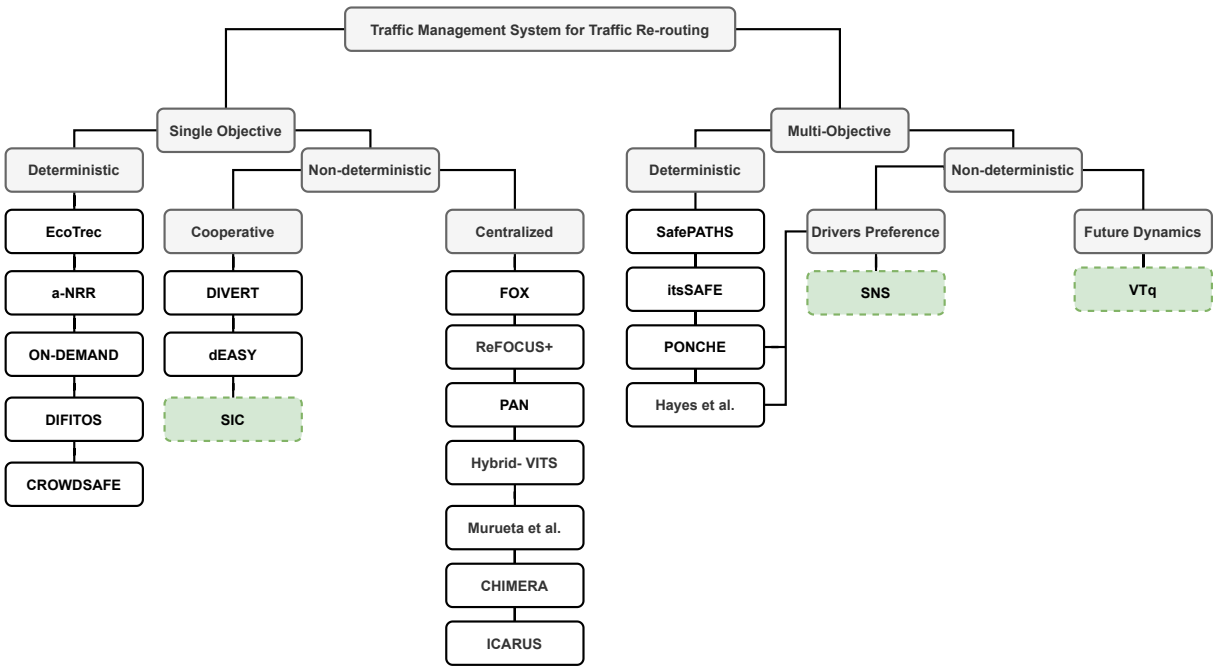


Figure 3.1: A TMS classification for traffic re-routing.

3.1 Solutions based on Single Objective Re-routing

Single objective re-routing solutions focus on improving the routing efficiency considering a particular urban aspect such as travel time, congestion index, fuel consumption, and even safety issues. Thus, these solutions tackle the following issue: how to re-route vehicles throughout the most efficient path using a specific urban aspect as a routing metric?

3.1.1 Deterministic Solutions

Deterministic solutions aim at improving the desired metric for each vehicle. However, depending on the number of vehicles and their origin and destination pairs, deterministic solutions potentially degrade re-routing efficiency. For instance, considering re-routing approaches that use the travel time as routing objective might create additional congestion spots because vehicles with similar origin and destination pair will be re-routed throughout the same path.

Doolan et al. [33] proposed EcoTrec, an eco-friendly TMS, which focuses on reducing carbon emissions while improving traffic efficiency. EcoTrec uses measurements periodically reported by vehicles to a central server to build a global traffic view. Each measurement is reported using an *Epidemic Routing* data dissemination protocol to ensure its delivery. Based on the measurements received by each vehicle, the central entity builds the traffic view and then forwards it to vehicles at a predefined interval. Thus, vehicles compute the optimal route based on the shortest path algorithm upon receiving a traffic view. Consequently, this reduces carbon emission and improves traffic efficiency since congested roads have higher fuel consumption rates. It is important to stress that whenever a vehicle receives an updated traffic view, its optimal route is re-calculated.

Wang et *al.* [65] introduce a-NRR, an adaptive next road re-routing system for unexpected urban traffic congestion avoidance. a-NRR saves the cost of obtaining global traffic condition knowledge by using local information available on RSU employed in each intersection to select the best next road rather than the whole route. Each vehicle broadcasts its status through beacon messages reported at least every 0.1 second. Also, to detect unexpected congestion, a-NRR relies on a *Traffic Operation Center* (TOC) that sends a notification to the RSU close to the traffic congestion, which broadcasts it to vehicles within its coverage. Upon receiving this notification, each vehicle can verify whether it will pass through the congested road. If necessary, it requests a detour for the RSU that uses the latest traffic information to compute the following route.

Gomides et *al.* [35] propose ON-DEMAND, an adaptive and distributed TMS using VANETS, which is based on V2V communication to estimate traffic congestion locally. To do so, it keeps track of its traveled distance in respect of the expected one when considering a free-flow traffic condition. Thus, the difference between these measurements is used to classify a contention factor, i.e., the vehicle perception on the road traffic condition. Each vehicle uses the contention factor to classify the overall congestion level, and this information is proactively disseminated to its vicinity considering an adaptive approach. If a vehicle does not have the necessary traffic information to estimate alternative routes, it discovers traffic information knowledge reactively. Moreover, each vehicle re-routes itself

based on the traffic estimation to improve the overall traffic efficiency using a deterministic shortest path algorithm.

Zhang et *al.* [69] proposed DIFTOS, a distributed infrastructure-less congestion prediction and traffic optimization system for VANETs in an urban environment. DIFTOS presents a hierarchical fashion design that aims to find each vehicle the shortest path from its source to the destination point. The first shortest path is computed based on the link travel delay with the weight constraint. After that, the algorithm determines a set of predicted congested road segments. Following that, a re-routing procedure is applied to each of these segments or sub-paths formed by the predicted congested road segments.

On the other hand, motivated by high criminal activities over the city, safety-based solutions have been proposed [60]. The key idea is to recommend the safest routes for all vehicles based on knowledge about dangerous areas and neighborhoods built from criminal incident reports, which can be obtained either from official criminal statistics [34] or social media participatory sensing systems [60]. Shah et al. [60] proposed CROWDSAFE seeking to handle traffic-safety concerns. CROWDSAFE is a crowdsourcing-based system to suggest the safest routes for the drivers. It allows people to report crime-related incidents to a central entity using their smartphone, and then it aggregates and refines reports to build knowledge about crime incidents. In this way, CROWDSAFE is capable of suggesting safer routes for users. Nevertheless, as its mechanism is based only on crowd reports, the acquired knowledge about crime incidents might be incomplete, once many crimes may not be reported at all, and there can be misreported.

3.1.2 Non-deterministic Solutions

Non-deterministic solutions reduce the problem of creating different congestion spots by distributing the traffic flow through a set of alternative routes. In other words, vehicles with similar origins and destinations potentially will be re-routed through different paths. For these solutions, two major designs are proposed: *(i)* centralized, in which a central server is responsible for distributed the traffic flow over a set of alternative routes; and *(ii)* cooperative, in which the vehicles themselves are responsible to balance the traffic flow cooperatively based on the pieces of information of its vicinity.

Centralized Design

Brennand et *al.* [10] introduce FOX, a fog-computing-based TMS that relies on V2I communications in order to optimize the traffic flow of the vehicles, focusing on minimizing traffic congestion and consequently reducing travel time, fuel consumption, and CO_2 emissions. FOX detects traffic congestion and controls the traffic of vehicles which is distributed over a set of RSUs that cover the entire scenario. Periodically, each RSU re-routes all vehicles within its coverage using the current position of the vehicles and the very last road of the current path of the vehicles under the coverage of the RSU as destination (i.e., the vehicles are only re-routed within the area of each RSU using local improvements instead of a global one). Re-routing is performed using a k Shortest Paths algorithm, in which each RSU computes a set of k possible routes. Thus, FOX uses a probabilistic approach to balance the traffic flow.

Re-routing with FOg-CloUd System (ReFOCUS+) [58] is another TMS to improve the overall traffic efficiency by re-routing vehicles. ReFOCUS+ uses RSUs to calculate traffic factors such as current and predicted congestion and travel time. Using this data, it applies re-routing to vehicles to try and reduce traffic congestion. The re-routing applied uses a multi-metric function called Road Weight Measurement (RWM), which means that many factors such as travel time, emissions, and distance are included in the calculations.

Murueta et *al* [56] propose a vehicle re-routing system to avoid the congestion that uses a model based on deep learning to predict the future state of the traffic network. The model uses the information obtained from the previous step to determine the zones with possible congestion and redirect the vehicles about to cross them. Alternative routes are generated using the entropy-balanced k Shortest Path algorithm (EBkSP). The proposal uses information obtained in real-time by a set of probe cars to detect non-recurrent congestion.

Pan et *al.* [53] describe traffic re-routing strategies designed to be incorporated in a cost-effective vehicular traffic guidance system. The re-routing computes alternatives routes using the following strategies: *(i)* Random k Shortest Paths (RkSP), balances the traffic by selecting a route randomly in a set of k possible routes. This strategy intends to reduce the possibility of creating different congestion spots. However, it can select long routes to the vehicles; and *(ii)* Entropy Balanced k Shortest Paths (EBkSP), which is an improvement of RkSP, in which a more intelligent route selection is made by considering the impact that each selection has on the future density of an affected road segment.

Ahmad et *al.* [1] describe route planning and data dissemination in real-time using the Internet of Things paradigm. A novel data dissemination technique achieves this objective for information sharing among the roadside units in a hybrid VANET intelligent transportation system (Hybrid- VITS). Hybrid-VITS comprises VANETs, vehicular traffic servers, and a 5G-based cellular system of public transportation. By considering traffic congestion in urban areas, the optimal path is calculated to replicate routes based on the k shortest path algorithm. A load balancing technique is adopted to avoid further congestion.

Souza et *al.* [29] introduce ICARUS (Improvement of Traffic Condition through an Alerting and Re-routing System), a TMS, which receives information about traffic hazards from other systems such as congestion detection, accident warnings, congestion prediction, bad weather conditions, etc. Upon receiving this information, a vehicle identifies traffic hazards and critical areas potentially affected. Thus, ICARUS creates an alert message and spread through an efficient data dissemination protocol based on V2V communications to warn vehicles that will approach those areas. After that, to each critical area, an area of interest is defined to warn just the vehicles within it to prioritize them and do not spread this information to vehicles that have no interest. Moreover, ICARUS acts both proactively and reactively. Vehicles are warned as soon as a traffic hazard occurs. Thus it can control traffic congestion first than other systems like [53, 22, 21, 33, 54, 65]. Furthermore, when a vehicle receives a warning message, it can verify if it will pass by a critical area. If necessary, it can request another route to a central server through V2I communication to improve overall traffic efficiency.

Cooperative Design

Pan et al. [54] introduced DIVERT, a hybrid TMS that computes the fastest routes cooperatively to each vehicle based on a global traffic view. DIVERT is composed of a central server responsible for detecting signs of congestion and software running on each car, which is responsible for reporting its traffic information (e.g., the travel time of each road) to the central server using cellular infrastructure to build the traffic view. Moreover, it also employs a vehicular networking-based information reporting protocol to minimize network contentions and reduce privacy leakage (e.g., the current location of several vehicles). For improving traffic efficiency, when traffic congestion is detected, the central server sends a notification message with the global traffic view to the vehicles close to it to reroute themselves to avoid the congestion. The rerouting algorithm is executed cooperatively to provide better traffic flow and prevent bottlenecks in the transportation infrastructure.

Akabane et *al.* [2] propose a distributed vehicle traffic management system, named as dEASY (distributed vEhicle trAffic management SYstem). dEASY system is designed based on a three-layer architecture, namely: environment sensing and vehicle ranking, knowledge generation and distribution, and knowledge consumption arranged as follows: the first layer deals with the task of selecting the most appropriate vehicle to perform data forwarding and/or knowledge generation, the second one addresses the knowledge generation and distribution, and the third layer applies an altruistic approach to choose an alternative route. dEASY is a fully distributed system, thus to perform the altruist re-routing, the vehicles need to balance the traffic flow cooperatively. However, vehicles are re-routed based on local knowledge about traffic conditions, potentially decreasing the overall re-routing efficiency.

3.2 Solutions based on Multi-Objective Re-routing

Multi-objective solutions focus on improving routing efficiency considering more than one urban aspect. However, different aspects potentially lead to different paths. Therefore the critical issue in these solutions is how to explore the trade-off among two or more urban aspects during the re-routing. For instance, the fastest route might not be the safest one. Thus these solutions handle the following issue: how to improve the traffic efficiency while also improving the safety of drivers and passengers?

3.2.1 Deterministic Solutions

Differently from [60], Galbrun et al. [34] introduced a system to suggest the safest routes for the users (e.g., pedestrian and drivers) based on knowledge obtained from an official dataset of criminal events, which is publicly available by an open government initiative of the U.S[1]. Moreover, considering the distance traveled by the users, Galbrun et al. employed a bi-objective algorithm to suggest safe paths, which is aware of both safety risks and distance, to recommend shorter paths, but which avoid dangerous areas. A

[1] `http://www.data.gov/about`

set of different paths (i.e., all possible paths) is computed, and the total distance and safety risk of each path is calculated so the best path can be determined. However, computing all paths can be too costly, potentially degrading system performance. Using only information about distance does not provide better traffic management due to a lack of knowledge about traffic conditions on roads.

Souza et *al.* [20] we proposed a TMS for improving SAfety and traFfic Efficiency named as itsSAFE, which: *(i)* employs accurate knowledge about the traffic conditions and unsafe levels on roads; *(ii)* and use these data to suggest periodically alternative routes for all vehicles. The problem is modeled as an instance of the Resource-Constrained Shortest Path (RCSP) and minimizes two objectives (traffic congestion and safety risk level). A dynamic programming routing algorithm is used to determine the most efficient route to a vehicle satisfying the safety requirements.

On the other hand, Hayes et *al.* [37] propose and test a personalized routing application that allows end-users to flexibly adjust their route preferences among travel distance, estimated travel time, and the safety level. Similarly, Ladeira et *al.* [42] propose PONCHE, a personalized TMS for traffic re-routing in which each driver's profile is reflected into contextual data type weights considered by the system, i.e., the intensity he/she wants to avoid a contextual region. With that, a driver's profile may ignore a determined contextual data type.

3.2.2 Non-deterministic Solutions

Non-deterministic solutions in multi-objective re-routing aim at improving several urban aspects without degrading the efficiency of each other. Unfortunately, until the time that this thesis was written, none non-deterministic multi-objective re-routing solution for TMS was found in the literature.

3.3 Analysis of Literature Solutions for Traffic Re-routing

This section analyses the literature solutions for traffic re-routing highlighting their limitations and drawbacks. Table 3.1 summarizes the solutions described in the previous section providing a qualitative comparison among them. The bold solutions represent the TMS developed in this thesis.

To analyze the literature solutions the following requirements will be used: *(i)* scalability; *(ii)* re-routing efficiency; *(iii)* personalization; and *(iv)* reliability. *Scalability* refers to the ability to deliver the desired performance without carrying about the number of vehicles or the size of the scenario. On the other hand, *re-routing efficiency* refers to how good is the traffic management of the solution. *Personalization* points out the tweaks available for the end-users (i.e., vehicles/drivers) considering multiple urban aspects and even user preference. Finally, *reliability* determines how reliable the system's routes are concerning future changes in the urban environment. Thus, the literature solutions will be analyzed considering these requirements

The major issue related to scalability is the fact that several solutions use a central server to re-route vehicles [33, 65, 69, 53, 58, 1, 56, 22, 29, 60], this design leads to the

Table 3.1:

	Architecture		Routing strategy		Optimization		Tweaks	
Related Work	Centralized	Distributed	Deterministic	Non-deterministic	Single-objective	Multi-objective	Personalization	Future Dynamics
EcoTrec[33]	√		√		√			
a-NRR[65]	√		√		√			
ON-DEMAND[35]		√	√		√			
DIFITOS[69]		√		√	√			
CROWDSAFE [60]	√		√		√			
DIVERT [54]		√		√	√			
dEASY [2]		√						
FOX [8]	√			√	√			
ReFOCUS+ [58]	√			√	√			
Pan et al. [53]	√			√	√			
Hybrid-VITS [1]	√			√	√			
Murueta et al. [56]	√			√	√			
CHIMERA [22]	√			√	√			
ICARUS [29]	√			√	√			
safePATHS [34]			√			√		
itsSAFE [20]			√			√		
PONCHE [42]			√			√	√	
Hayes et al. [37]			√			√	√	
SIC		√		√	√			
SNS		√		√		√	√	
VTq		√		√		√	√	√

following problems: *(i)* network contentions, which is because every vehicle in the network needs to report traffic-related information to the server and also request alternative routes, consequently increasing the communication between vehicles and RSUs, which can be a bottleneck depending on the number of vehicles and the bandwidth available [67]; and *(ii)* undesired latency, depending on the complexity of the re-routing algorithm the number of vehicles to be re-routed and the size of the scenario might introduce latency to the system, consequently degrading the overall traffic efficiency.

Fully-distributed solutions were proposed to deal with this problem, such as ON-DEMAND [35] and dEASY [2]. The key idea of these solutions is to provide efficient data sharing protocols to create local knowledge about traffic conditions in each vehicle. Then the vehicles are responsible for re-routing themselves. However, the critical issue related to these solutions is that they rely on local knowledge about traffic conditions, which potentially decreases the re-routing efficiency regarding solutions that consider the global knowledge about traffic conditions. To handle such a problem, DIVERT [54] proposes a hybrid approach to create global knowledge about traffic conditions. Besides, to enable efficient traffic management DIVERT proposes a cooperative re-routing algorithm to balance the traffic flow. However, DIVERT implements a probabilistic algorithm to determine the set of vehicles to report traffic-related information, which is not the most efficient way to create an accurate traffic view [30].

The drawbacks related to the routing efficiency are expressed according to the design of the re-routing algorithm, in which deterministic solutions potentially decrease the routing efficiency by creating bottlenecks as a consequence of the re-routing which is the case of the solutions [33, 65, 35, 60, 42, 34, 20, 69, 37]. On the other hand, non-deterministic approaches reduce such a problem by providing a better balance of the traffic flow over a set of alternative routes which is the case of the following solutions [54, 2, 53, 1, 22, 29, 56, 58]. Nevertheless, to enable efficient and real-time re-routing, the system must have a collaborative design to avoid issues related to scalability such as DIVERT [54] and dEASY [2].

Many urban aspects might impact re-routing decisions from a user's point of view. Thus, different users with different preferences might prefer different routes. However,

few solutions have a multi-objective design and provide the desired personalization, such as PONCHE [42] and Hayes et *al.* [37]. Nevertheless, despite the multi-objective features, these solutions still have a deterministic approach to re-route vehicles, which leads to the same problems as the single objective ones. In addition, none of the presented solutions care about future changes in the dynamics of urban aspects, consequently decreasing the reliability of the route recommended by the system since even a tiny change in the dynamics might decrease the routing efficiency. These solutions rely on frequent re-routing. However, this approach increases not only the network costs, but also computation efforts because the system needs to gather the traffic information of all vehicles in the network and compute new routes to improve the overall traffic efficiency in each re-routing phase.

With this analysis, it is possible to conclude that an efficiency TMS for traffic management needs to have: *(i)* a scalable architecture to enable an efficient and real-time re-routing; *(ii)* a multi-objective non-deterministic re-routing algorithm to enable the personalization according to the preference of each user to improve the traffic management considering several urban aspects without degrading the efficiency of each other; and *(iii)* a re-routing algorithm that cares about future changes in urban dynamics to improve the reliability of the route computed by the system.

3.4 Chapter Conclusions

This chapter described literature solutions for the traffic management problem, highlighting advantages and limitations for each solution. Also, a classification and a qualitative comparison were proposed. In summary, the literature solutions presented have limitations related to: *(i)* system scalability; *(ii)* multi-objective and personalized re-routing; and *(iii)* lack of knowledge about future changes in the urban dynamics. In this way, the next chapters will describe the solutions proposed to address these limitations. Therefore Chapter 4 presents the solution for dealing with scalability issues and also enables real-time re-routing. Chapter 5 describes the solutions for dealing with the multi-objective and personalized re-routing to improve the traffic management considering multiple urban aspects without degrading the efficiency of each other. At last, Chapter 6 introduces a solution for dealing with future changes in the urban dynamics during the route planning to improve the reliability of the routes recommended by the system.

Chapter 4

Towards a Scalable Cooperative Vehicular Traffic Re-Routing System

Research Question 1: *How to ensure system scalability with low overhead and reasonable traffic management?*

Vehicular traffic re-routing is the key to handling mobility problems by suggesting alternative routes to improve overall traffic efficiency. Also, it is possible to increase the road traffic flow without increasing and changing the road infrastructure. The critical idea of traffic re-routing systems is to create accurate knowledge about traffic conditions based on vehicles' traffic information. Then, providing an efficient routing mechanism to balance the traffic flow without creating additional congestion spots.

There are at least three issues that need to be handled to provide an efficient traffic re-routing system, including *(i)* network overhead; *(ii)* scalability; and *(iii)* routing efficiency. First, the system needs to provide efficient traffic reporting mechanisms that do not overload the network and still enable accurate traffic views. Second, the system needs to provide mechanisms to enable real-time re-routing without introducing undesired latency to the system. Finally, the routing algorithm needs to be aware of the routes taken by other vehicles to balance the traffic flow properly. In this way, this chapter proposes Sharing is Caring (SIC), an efficient cooperative traffic re-routing system. SIC implements efficient cooperative mechanisms to create an accurate traffic view and re-route vehicles efficiently.

By the end of this chapter, we will have shown solutions to reduce the following issues: *(i)* how to reduce the number of transmissions and still provide accurate knowledge about traffic conditions; *(ii)* how to re-route vehicles in a distributed and cooperative manner in which the vehicles are aware of the routes taken by their neighbors to improve the overall traffic efficiency; and *(iii)* how to compress the data disseminated through the network to avoid contentions.

The rest of the chapter is organized as follows. Section 4.1 presents the system overview. Section 4.2 introduces the problem statement for vehicular traffic re-routing. Section 4.3 describes the traffic reporting mechanism implemented by SIC, while Section 4.4 describes the cooperative re-routing algorithm. Section 4.5 presents the optimizations to improve the network efficiency of the system. At last, Section 4.6 shows the performance analysis and Section 4.7 concludes the chapter.

4.1 System Overview

SIC is based on two design principles, which are related to the following concerns: *(i)* real-time vehicular traffic re-routing; and *(ii)* network contention minimization. First, the re-routing algorithm must be offloaded from the cloud server to each vehicle to reduce the computation time and the communication burden on the server, consequently providing better scalability to the system. Also, the re-routing algorithm must be collaborative. In other words, each vehicle needs to be aware of the routes taken by its neighbors to avoid the creation of additional congestion spots and to achieve a better re-routing effectiveness. Finally, the communication between vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) must be optimized to increase the delivery ratio and coverage while reducing the overhead of messages and network contentions.

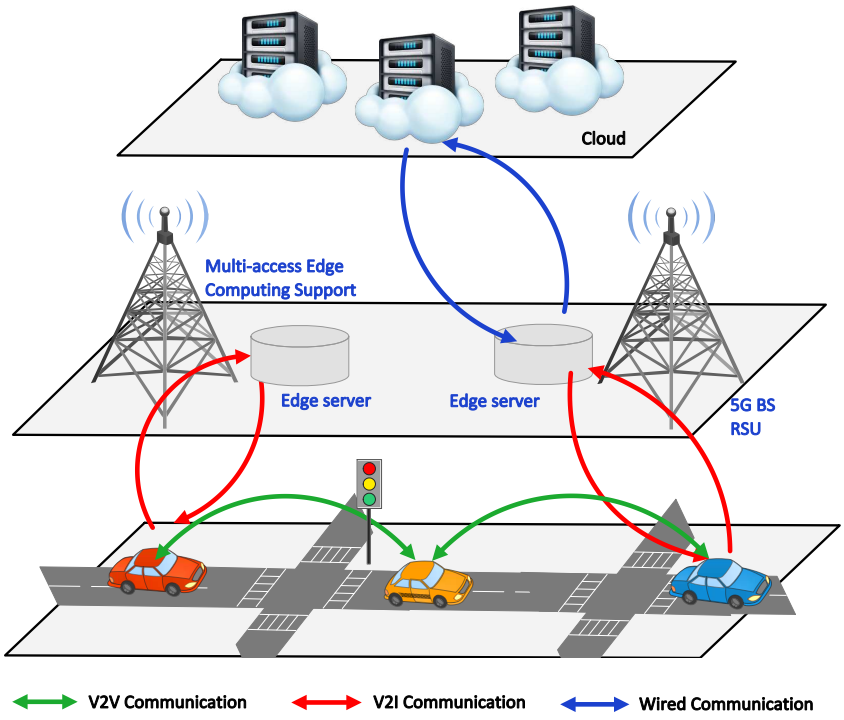


Figure 4.1: SIC's architecture

Figure 4.1 depicts the system architecture. SIC employs a MEC-based architecture composed of vehicles, roadside infrastructures (i.e., RSUs, 5G base stations, access points), edge servers, and the cloud. Vehicles equipped with on-board units (OBU) communicate with other vehicles (i.e., V2V communication) to sense the urban environment (e.g., estimate travel time, velocity, fuel consumption, and CO_2 emission on the roads). Also, vehicles communicate with roadside infrastructures, edge servers, and with the remote cloud over either 5G or vehicular networking (i.e., V2I communication) to report traffic-related data and to request/receive pieces of information. The edge servers are widely deployed on roadside infrastructures to provide processing and storage resources on the network edge, enabling fast responsiveness.

In SIC, each edge server is responsible for building local knowledge about traffic conditions over its coverage area based on vehicles' reports. Each edge server can also provide

intensive computation tasks such as traffic and congestion prediction to improve vehicular traffic management. On the other hand, the remote cloud is responsible for building global knowledge about traffic conditions and dealing with edge servers' resource constraints. Thus, SIC can build knowledge about traffic conditions to detect and predict congested roads. Whenever traffic congestion is detected, the server (deployed in the cloud or edge servers) can notify the vehicles and send the traffic view to them to enable that each vehicle computes its route cooperatively (i.e., computation offloading) to improve the overall mobility.

4.2 Problem Statement

The road network is represented by a direct graph $G = (V, E)$, in which the set of vertices V represents the road intersections, while the set $E \subseteq V \times V$ corresponds to the road segments, i.e., the road segment $uv \in E$ represents the road segment connecting the intersections u and v. Each road segment $uv \in E$ has a length represented by l_{uv} and a traffic condition estimation at time t_i represented by $\tau_{uv}^{t_i}$. Every vehicle in the network is represented by the set N. Thus, each vehicle $n \in N$ has a pair of origin $s \in V$ and destination $t \in V$, such that $s \neq t$, which is associated to a path $P \subseteq E$ connecting $s \rightarrow t$, defining the vehicles' route at time t_i. The traffic efficiency $\tau_P^{t_i}$ of a path P at time t_i is defined as:

$$\tau_P^{t_i} = \sum_{uv \ \in P} \tau_{uv}^{t_i} \tag{4.1}$$

Thus, let x_{uv} be the variable that defines P, such that:

$$x_{uv} = \begin{cases} 1 & \text{if } uv \in P \\ 0 & Otherwise \end{cases} \tag{4.2}$$

In this way, the problem of improving the overall traffic efficiency is finding a new path P at time t_i considering that $\tau_P^{t_i} < \tau_P^{t_{i-1}}$ for each vehicle based on the ongoing traffic condition estimations at time t_i and its origin s and destination t. Formally, this problem can be described as:

$$\min \sum_{uv \in E} \tau_{uv}^{t_i} x_{uv} \tag{4.3}$$

subject to:

$$\sum_{uv \ \in \text{ out}(v)} x_{uv} - \sum_{vu \ \in \text{ in}(v)} x_{vu} = \begin{cases} 1 & \text{if } v = s \\ -1 & \text{if } v = t \\ 0 & \text{if } v \neq s \text{ and } v \neq t \end{cases} \tag{4.4}$$

$$x_{uv} \in \{0, 1\} \ \forall uv \in E \tag{4.5}$$

4.3 Vehicular Traffic-aware Data Reporting

Vehicles need to sense the urban environment, i.e., the number of vehicles, velocity, travel time on the roads, and frequently report traffic-related information to enable accurate traffic estimations and allow timely congestion detection and decision-making. However, whether all vehicles try to report their traffic information at the same time, they can produce network contentions, which will degrade the overall system performance [54]. Therefore, SIC focuses on minimizing traffic information reported by the vehicles while producing accurate knowledge about traffic conditions. Finally, we present a traffic condition estimation algorithm based on the traffic reports.

4.3.1 Traffic Reporting

Vehicular traffic-aware reporting is based on two assumptions: *(i)* vehicles need to inform which roads are not with free-flow traffic condition, and *(ii)* vehicles in denser areas are more likely to report redundant information. In this way, we propose a density-based mechanism with vehicles reporting their traffic information only when the density is higher than a predefined threshold. This approach reduces overhead, and it does not degrade the re-routing effectiveness since SIC can still detect congestion properly.

The main idea of this approach is to minimize the number of traffic reports by partitioning the road network into smaller regions named sub-regions. Hereafter, a vehicle is selected to report its traffic estimation in each sub-region. Road network partitioning is a trade-off between the accuracy of knowledge about traffic conditions and the number of traffic reports. Thus, the greater the sub-regions size, the lower the number of traffic information reported to the central server. However, each sub-region must be defined carefully because the larger its size is, the harder it will be to estimate its traffic conditions accurately [30]. In order to find the most appropriate size and shape of each sub-region, SIC employs the k-means clustering algorithm [67].

The number of sub-regions is defined to enable cooperative traffic sensing, focusing on maximizing the coverage area with as few sub-regions as possible based on the communication range of the vehicles. Therefore, the number of sub-regions κ is defined as:

$$\kappa = \lceil \frac{\text{road network area}}{2\pi \cdot r_N^2} \rceil \tag{4.6}$$

where r_n is the communication range of the vehicles. In summary, we want to find how many circles (based on the communication range of the vehicles) are necessary to cover the entire road network area.

The k-means algorithm is based on an interleaving approach, where the cluster assignments $V_k \subseteq V$ for $k \in \{1, 2, \ldots, \kappa\}$ are established given the centers which are computed given the assignments. The optimization criterion is as follows:

$$\min_{V_k} \sum_{k \in \kappa} \sum_{v \in V_k} \parallel v - c_k \parallel^2 \tag{4.7}$$

where $\parallel v - c_k \parallel^2$ is the distance between vertex v and centroid c_k, while V_k is the subset

of vertices associated to the sub-region k and κ is total number of sub-regions. In this way, each sub-region is defined as $E_k \subseteq E$ for each $uv \in E_k$ such that $u, v \in V_k$ for each $k \in \{1, 2, \ldots, \kappa\}$.

After defining each sub-region, the traffic information is reported according to the following requirements: *(i)* the traffic information is sent when a vehicle verifies that the density of a particular road in its sub-region can potentially evolve to congestion and *(ii)* SIC chooses which vehicle will be responsible for reporting the traffic estimation by using a delay-based approach as a function of the density on its sub-region.

The estimated density is computed locally by every vehicle in the sub-region based on the neighborhood's information (i.e., within the same sub-region) using periodic beacons exchange. Each vehicle emits beacons with the same frequency. Then, by counting the number of received beacons in a short time window (e.g., 5 seconds), each vehicle can estimate the density of its vicinity. Beacons consist of a tuple $\langle timestamp, n, uv, velocity \rangle$, representing the timestamp, the vehicle identification $n \in N$, the road segment $uv \in E$ and the velocity of the vehicle.

The delay-based approach enables every vehicle in the same sub-region to schedule a traffic reporting based on a computed delay. To avoid redundant transmission, if a vehicle overhears another report from the same sub-region during its waiting time (i.e., according to its delay), it cancels its reporting. The delay is computed based on the degree of each vehicle (e.g., number of neighbors), defined as:

$$d_n = -\left(\frac{max(d)}{max(\delta)^{\varepsilon}}\right) \cdot \delta(n) + max(d), \tag{4.8}$$

where $\delta(n)$ is the neighbor degree of vehicle $n \in N$, while $max(\delta)$ is the maximum neighbor degree of its sub-region, which is estimated based on the received beacons, and $max(d)$ is a parameterized value to define the maximum delay to report traffic estimation to the main server.

This approach considers the number of neighbors to prioritize vehicles with better estimations. Since our implementation, the sub-regions have about the same size as the communication range of the vehicles; the vehicles with a higher degree potentially will have a better traffic estimation than the others. For the sake of clarity, Figure 4.2 shows the previously described delay-based approach to reduce the number of traffic report transmissions. After estimating the traffic condition of the roads under its coverage, each vehicle schedules a traffic report using the delay-based approach (considering the number of vehicles in its vicinity). As it can be seen in Figure 4.2, the vehicle close to the center of each sub-region has a higher degree (i.e., vehicles in its vicinity) than the others. Thus, they are responsible for sending their traffic estimations to the server. At last, when the other vehicles detect this communication, they will cancel their transmission to avoid redundant transmissions.

4.3.2 Traffic Condition Estimation

The server receives the reports from vehicles containing the number of vehicles on each road segment for every road on their sub-regions. Whenever the server receives a report

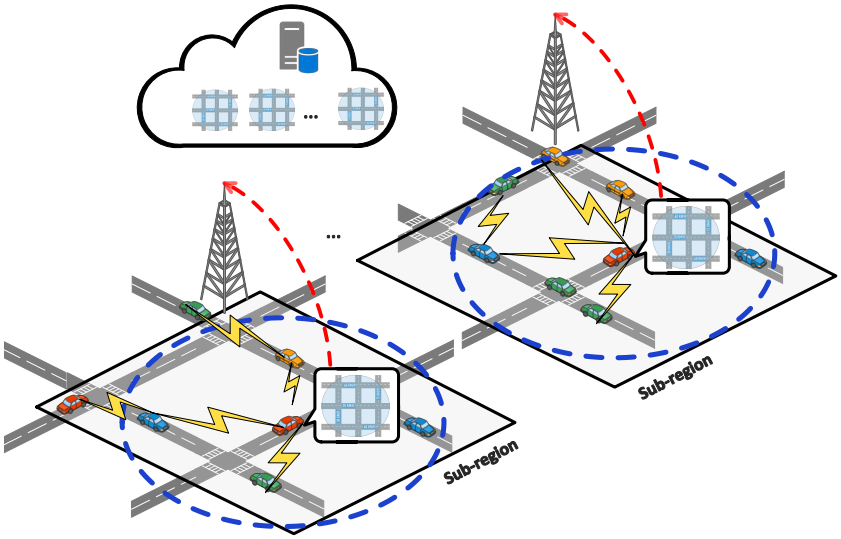


Figure 4.2: Delay-based approach considering traffic density and the sub-regions

concerning a sub-region, it will compute an exponential moving average [54] for each road in that sub-region. In this sense, the server estimates the traffic condition considering two cases:

Free-flow estimation: for the road segments without any traffic reports, the estimation is considered as free-flow, which is equal to the travel time spent to travel the entire road segment.

Traffic condition estimation: for the roads with a density greater than 0, the traffic condition estimation is based on the Greenshield model [6], which considers the relation between speed and density. This model is extensively used by transportation researchers and was shown empirically to describe well the speed-density relation for relatively low densities. It considers a linear relationship between the estimated average speed and the traffic density on each road segment as follows:

$$avgs_{uv}^{t_i} = maxs_{uv} \left(1 - \frac{N_{uv}^{t_i}}{\frac{l_{uv}}{len(n)+\beta} \cdot lanes} \right) \tag{4.9}$$

$$\tau_{uv}^{t_i} = \frac{l_{uv}}{avgs_{uv}^{t_i}}, \tag{4.10}$$

where, $N_{uv}^{t_i}$ is the current vehicular density of road uv at time t_i, $\frac{l_{uv}}{len(n)+\beta}$ is the maximum density of the road uv computed based on the length of the road l_{uv}, the average vehicles' length $len(n)$, the minimum gap β between each vehicle in each lane of the road uv and the number of lanes of such road. The average speed $avgs_{uv}^{t_i}$ at time t_i is estimated according to the model and considering its maximum speed $maxs_{uv}$.

4.4 Cooperative Distributed Re-routing Approach

When the server detects signs of congestion in any road, it will alert the vehicles by sending the updated traffic view (e.g., current traffic conditions on the roads) containing all roads with traffic conditions different from the last update. The server sends the traffic view only to vehicles that reported the sub-region traffic estimation most recently and are closer to the congestion spots. The traffic view update triggers the re-routing processes, which is composed of two main phases: *(i)* Traffic view dissemination; and *(ii)* Cooperative route computation. The first phase is concerned with providing the updated traffic view to the vehicles that potentially will be affected by the congestion. Simultaneously, the other one is responsible for optimizing the re-routing algorithm to achieve better traffic management.

A vehicular traffic re-routing algorithm's effectiveness is directly related to its ability to balance the traffic flow to avoid the creation of different congestion spots. When computing an alternative route to some vehicle, the re-routing algorithm must be aware of the routes previously computed by the other vehicles in the same time window (i.e., re-routing interval). However, in distributed approaches, where each vehicle computes its alternative route, this task is not as straightforward as in centralized ones. Each vehicle needs to inform the others about its route whenever a new route is computed in distributed approaches. Nevertheless, spreading all routes of all vehicles in a multi-hop approach to the whole network during every re-routing phase is not feasible since it will produce a high network contention as the density of vehicles increases. Besides, since each vehicle needs to wait for the other vehicles' routes to compute its route, it might introduce high latency to the system if not adequately addressed.

In this scenario, the distributed re-routing algorithm needs to consider the following issues: *(i)* how to properly notify vehicles about the computed routes; and *(ii)* how to balance the traffic based on the information of its neighborhood.

4.4.1 Cooperative Route Sharing

The re-routing algorithm proposed in SIC focuses on balancing the traffic flow over a set of alternative routes for each vehicle, computed using the K-Shortest Paths algorithm [68]. However, to effectively provide this approach in a distributed way, the vehicles need to know as many routes taken by their neighbors as possible to achieve better route guidance. Also, they cannot wait to compute their routes after all their neighbors because it potentially introduces an undesired delay in the re-routing process.

To tackle these concerns, we propose a rank-based approach that gives a higher priority to vehicles closer to the congestion to compute their routes before the others, assuming that those vehicles are the ones that need a faster response. Rank is a delay calculated as the function of the distance between the vehicle and the next congested road in its route. This rank-based approach minimizes the broadcast storm problem during route sharing since it avoids that vehicles share their routes simultaneously. Also, it reduces the latency because vehicles only wait for the routes from vehicles with higher ranks rather than all their neighbors.

Algorithm 1: Cooperative route sharing algorithm

```
   Input  : G // Traffic view built by the server
          1 C // Set of congested roads detected by the server
          2 K // Set of alternative routes for each vehicle
   Output: Broadcast of the set of alternatives paths of the vehicle and its priority
   // Update the knowledge about traffic conditions on the roads and disseminate it through the
      network using a data dissemination protocol
 3 updateTrafficCondition(G);
 4 disseminate(G);
   // Get the current route of the vehicle
 5 P ← getRoute();
   // Check if the vehicle will pass by a congestion road
 6 if ∃uv ∈ P | uv ∈ C then
       // Get the origin and destination of the vehicle
 7     s ← P.getCurrentRoad();
 8     t ← P.getDestination();
       // Compute K alternative routes from s to t
 9     P ← G.getKShortestPaths(s, t, K);
       // Compute priority based on the distance to the nearest congested road
10     rank ← computeRank(P, C);
       // wait rank ms to broadcast its alternatives paths and its priority
11     wait(rank);
12     compress(P);
13     broadcast(P, rank);
14 end
```

Algorithm 1 describes the cooperative routing sharing approach. In summary, when a vehicle receives some new information about traffic conditions, it updates its traffic knowledge (Line 3) and forwards it using a data dissemination protocol, described in Subsection 4.5.1. Then, the vehicle checks if it will pass by some congested road (Lines 5-6). If so, it computes the set of alternative routes $\mathcal{P}$ based on its current position, destination and the parameter K. Also, it computes its *rank* based on the distance between its current position and the nearest congested road $uv \in C \subseteq E$ (Lines 7-10). Finally, according to its rank, the vehicle waits to broadcast its set of alternative routes and its rank to its neighbors (Line 13).

It is important to notice that each vehicle shares its set of alternative routes and its rank to enable that other vehicles infer which route each vehicle has selected based on all routes received and the rank of each neighbor. The size of the shared message is proportional to the size of the paths and the number of alternative routes (i.e., the size of K). Consequently, in large scenarios, the payload of the shared messages will increase, thus introducing an undesired overhead to the system. Thus, a data compression mechanism is proposed in Subsection 4.5.2 to address such an issue.

4.4.2 Cooperative Re-routing Algorithm

SIC balances the traffic according to a probabilistic approach using a weighted popularity index to avoid creating different congestion spots. In summary, the probabilistic approach computes high probabilities to the least popular routes.

Definition: *The weighted popularity index of a road pop*(uv) *is defined by the number of times that a set of vehicles uses the road segment to compose its route. Given a vehicle, let* $\mid P_{uv} \mid$ *be the number of times that the road uv was selected to compose a route of its*

neighbors. Hence, we compute the weighted popularity of uv as:

$$pop(uv) = \mid P_{uv} \mid \cdot \left(1 - \frac{avgs(uv)}{maxs(uv)}\right), \tag{4.11}$$

We used the Boltzmann algorithm [41] as the probabilistic approach. Let $\mathcal{P}$ be the set of alternative paths and let k and T be the two parameters used by the Boltzmann algorithm, in which the first one represents the Boltzmann constant and the other one the parameterized attribute. In this way, each vehicle computes its Boltzmann constant according to:

$$k(P) = \sum_{uv \in P} \exp\left(-\frac{pop(uv)}{T}\right), \tag{4.12}$$

where P is a possible alternative path in the set $\mathcal{P}$, $pop(uv)$ is the weighted popularity index of each road in the path, and T is the parameterized value of the Boltzmann algorithm. Notice that the greater the value of T is, the higher the chance of achieving a uniform distribution, which must be avoided to provide a better path selection (e.g., avoid creating different congestion spots).

After computing the Boltzmann constant, each vehicle is able to compute the selection probability of each $P \in \mathcal{P}$ based on the popularity of its whole path. The probability is defined by $Pr(P)$:

$$Pr(P) = \frac{1}{\exp\left(\frac{\sum_{uv \in P} pop(uv)}{k(P) \cdot T}\right)}, \tag{4.13}$$

where P is a possible alternative path and $\sum_{uv \in P} pop(uv)$ its popularity.

Finally, based on the probability of each path, the vehicle is able to choose its alternative route P' using a random procedure based on each probability. The key idea is to select the path with the highest probability that satisfies the following condition:

$$P' = \underset{P \in \mathcal{P}}{\arg\max} \Big\{ X \cdot Pr(P), X \in [0, 1] \Big\}, \tag{4.14}$$

where X is a random variable to represent the random procedure and can assume values between 0 and 1.

For the sake of clarity, Algorithm 2 describes the procedure a vehicle executes to select its route based on its neighborhood information. Each vehicle first sorts the set of routes received from its neighborhood according to each rank (Line 2). Then, it compares its rank with the others and to each vehicle with a higher priority, it infers the route selected by the vehicle using the current popularity information and the Equation 5.9 (Lines 7-8). Thereafter, it updates the popularity $pop(uv)$ of each road $uv \in P$ in the route selected by the vehicle (Line 9). Finally, after inferring the routes taken by its neighbors with higher ranks, the vehicle computes its route avoiding popular routes (Lines 12-14).

Algorithm 2: Updating alternative route.

```
   Input  : G // Traffic view built by the server
            Q // The set of messages containing the alternative routes and the rank of each neighbor
   Output:
   // Sort Q according to the rank of each vehicle
 2 Q.sort();
   // Get its own rank
 3 r ← getMyRank();
 4 while Q is not empty do
 5 |   q ← Q.pop();
   |   // Check if its neighbor has a greater rank
 6 |   if q.rank < r then
   |   |   // Get the set alternative paths of its neighbor
 7 |   |   P ← q.getRoutes();
   |   |   // Infer the route taken based on the current traffic knowledge
 8 |   |   P ← selectRoute(P, G);
   |   |   // Update the traffic knowledge
 9 |   |   G.update(P);
10 |   end
11 end
   // Compute the least popular route based on the pieces of information shared by its neighbors
12 P ← myAlternativePaths();
13 P' ← selectRoute(P, G);
14 setRoute(P');
```

4.5 Vehicular Networking Optimization For Data Dissemination and Cooperative Route Sharing

To disseminate the traffic view, SIC uses vehicular networking. Thus, it needs to address the communication issues inherent to the vehicular environment, such as the broadcast storm problem, which appears when many vehicles close to each other try to transmit some information in a short time window [31, 54]. Also, it needs to employ a compression mechanism to reduce the size of the data packet payload containing the alternative routes of each vehicle.

4.5.1 Broadcast Storm

SIC employs a distance-based data protocol to avoid the broadcast storm problem. The main idea of this protocol is to prioritize farther away vehicles to continue the dissemination procedure. In this way, when a vehicle broadcasts the traffic view, all vehicles that receive the message schedule a re-transmission based on the inverse Euclidean distance between them and the vehicle transmitting the message. Therefore, farther away vehicles will re-transmit earlier than vehicles closer to the vehicle transmitting the message.

When a re-transmission is received, the vehicles that had already scheduled it can cancel their schedule to avoid a broadcast storm. Consequently, such an approach reduces the number of redundant transmissions. Detailed information about the data dissemination protocol is presented in [29].

Algorithm 3: Alternative routes compression algorithm

```
   Input  : 𝒫 // Set of alternative routes
   Output: Set of alternative routes compressed
   // List to store the compressed paths
1  CP ← [];
   // Get the shortest path in P
2  P ← 𝒫.getMostOverlappingPath();
3  for P′ ∈ 𝒫 \ P do
       // string used to compress the path
4      bits ← ∅;
       // list to store the path compressed
5      P_c ← [];
       // index iterator
6      idx ← 0;
7      for uv ∈ P′ do
           // Verify if the road segment is in the path P at index idx
8          if P′_idx equals to uv then
               // If so, add 0 to the bitstream to inform that that road is equal to the path P′ at
                  index idx
9              bits ← bits ∪ 0;
10             idx ← idx + 1;
11         end
12         else
               // Otherwise, add 1 to the bitstream to inform that that road is not equals to the
                  path P′ at index idx and add its related road to the end of the compressed path P_c
13             bits ← bits ∪ 1;
14             P_c.append(uv);
15         end
16     end
       // Add the bitstream to the first position of the compressed path P_c
17     P_c.inset(0, bits);
       // Add the compressed path P to the compressed paths list
18     CP.append(P_c);
19 end
20 return CP;
```

4.5.2 Alternative Routes Compression

The size of the data packet sent in the cooperative route sharing depends on the length and the number of alternative paths (i.e., parameter K of the K-Shortest Paths Algorithm). Thus, in high-density scenarios with several vehicles, many alternative routes with an extended traveling distance can potentially overload the network during the alternative route sharing. A large packet size increases the communication overhead and decreases the data dissemination effectiveness. Also, the MAC layer limits the payload of a data packet to be sent on the communication channel. In other words, the larger the data packet is, the higher is the number of data transmitted.

A data compression mechanism is proposed to tackle this problem. The idea is to compress the alternative paths around the most overlapping one using a bit representation, which is presented in Algorithm 3. First, the algorithm extracts the most overlapping path P from the set of alternative routes $\mathcal{P}$ (Line 2). Then, to each path $P' \in \mathcal{P} \setminus P$, a bit representation and a compressed version of the path is defined (Lines 3-6). In this way, for each road uv of the path P', the algorithm verifies if the road segment of P_{idx} at position idx is equal to the current road $uv \in P$ (Lines 7-8). If so, a bit 0 is added to the bit representation to inform that at position idx the current path P' has the same road of the path P' (Lines 8-10). Otherwise, a bit 1 is added to the bit representation to inform that it is different. Thus, the real road uv is appended to the compressed path representation

P_c (Lines 12-14). The real road is appended to the compressed representation to enable the decompression. When $uv \in P'$ is equal to the destination, the resulting compressed representation is added to the compressed paths CP. Upon finishing the compression of all paths, the vehicle shares its compressed version with its neighbors.

When a vehicle receives a compressed message and performs the decompression, it verifies the most overlapping path. To each path represented by its compressed version, it checks if the bit representation at position idx is equal 0. If so, the road uv of the most overlapping path is added to the original route. Otherwise, the vehicle adds the first unused road appended to its compressed representation.

4.6 Performance Analysis

This section analyzes the SIC's performance. Subsection 4.6.1 introduces the simulation platform, presenting the tools, scenario, while Subsection 4.6.2 describes the assessed metrics. Subsection 4.6.3 and Subsection 4.6.4 evaluate hyperparameters used by the k-means and by the k-shortest paths algorithms implemented in SIC. Subsection 4.6.5 evaluates the SIC's system performance compared to CHIMERA [22], DIVERT [54] and EcoTrec [33] respectively. CHIMERA is one of our previous works that employ the same re-routing approach. However, it has a centralized architecture to perform vehicular traffic re-routing. Thus, we want to evaluate how the proposed traffic-aware reporting and cooperative re-routing mechanisms impact both network performance and traffic efficiency with this analysis.

4.6.1 Methodology

The simulation platform is composed of the simulator of urban mobility, SUMO [7], version 0.30.0, the network simulator OMNeT++ [63], version 5.0 and also the vehicular networking framework Veins [61], version 4.6. The road network comprises a fragment of 5 km^2 from São Paulo city, Brazil, obtained using OpenStreetMap [36]. The traffic mobility was produced using the TrafficModeler [55] tool to ensure realistic traffic mobility, resulting in a total of five thousand routes. The number of routes was defined to create heavy traffic and congestion (i.e., the travel time for most cars is significantly higher than the free-flow travel time). In this way, we generate the traffic at a constant rate by deploying one car each second in the simulator from one side of the scenario to another. By default, the shortest travel time paths are automatically calculated and assigned to each vehicle at the beginning of the simulation based on the road speed limits. The presented results have a confidence interval of 95%, and Table 5.2 shows additional parameters used in the simulation.

4.6.2 Metrics

To evaluate SIC's performance and to compare with literature solutions, the following metrics were assessed:

Table 4.1: Simulation parameters

Parameters	**Values**
Channel frequency	5.890e9 Hz
Propagation model	*Two ray*
Transmission power	2.2 mW
Communication range	300 m
Bit rate	18 Mbit/s
PHY model	IEEE 801.11p
MAC model	EDCA
Max hop count	10
Scenario	São Paulo, Brazil
Scenario size	$5km^2$
Vehicles density	125, 250, 500, and 1000 vehicles/km^2
Compliance rate	25%, 50%, 75%, and 100%
# k-paths	1, 3, and 7 shortest paths
Re-routing interval	450 s

- **Transmitted messages per routing step** measures the number of messages transmitted by all vehicles in each route step to build and share the knowledge about traffic conditions (i.e., traffic view);

- **Total of transmitted messages** is the total number of messages transmitted by the system to build the traffic knowledge and re-route the vehicles;

- **Accuracy of the traffic view per vehicle** measures how accurate is the knowledge about the traffic conditions of each vehicle. The metric is expressed as a cumulative distribution function.

- **Average Travel Time** evaluates the efficiency of the routing algorithm. It is obtained based on each vehicle's average total time to travel its entire route.

- **CPU time** is the time spent by the re-routing algorithm to compute the new alternative route to the intended vehicles. A longer CPU time potentially introduces latency to the system, consequently degrading its overall performance.

- **System degradation** analyzes how the system is degraded. It is computed as the ratio of the traffic density and the number of transmitted messages times the accuracy of the traffic knowledge. The lower the values are, the lower the is the degradation of the system.

- **Data compression** measures the packet compression of the proposed algorithm to summarize the number of paths (i.e., k-paths) shared by the vehicles to enable the traffic balancing. The transmitted packet size is represented in *bytes*.

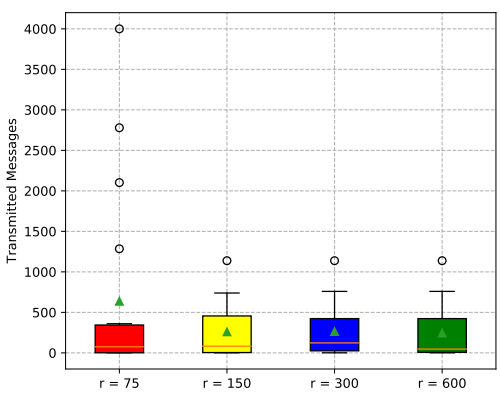
- **Penetration rate** represents the percentage (i.e.,25%, 50%, 75%, and 100%) of vehicles that will report the traffic information will follow the alternative computed route. This metric measures the system performance, considering the scenario in which fewer vehicles use SIC.

- **Re-routing frequency** is the total number of times that each vehicle was re-routed to perform its entire trip. This metric measures the driving experience since many

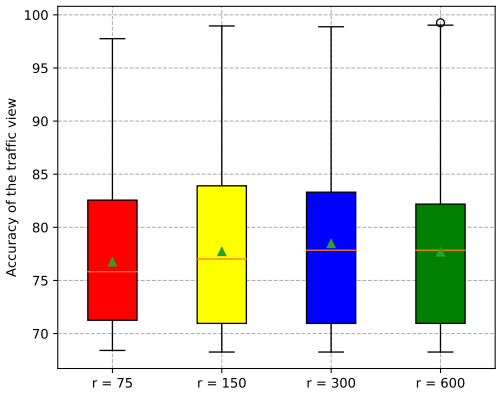
route changes throughout the trip potentially decrease the quality of the driving experience.

- **Route similarity** is the similarity between the route traveled and the route initially planned by each vehicle. This metric measures the quality of the alternative routes suggested by the system; it is important to highlight that high re-routing frequency with high similarity potentially means that the alternative recommended routes were not worth it.

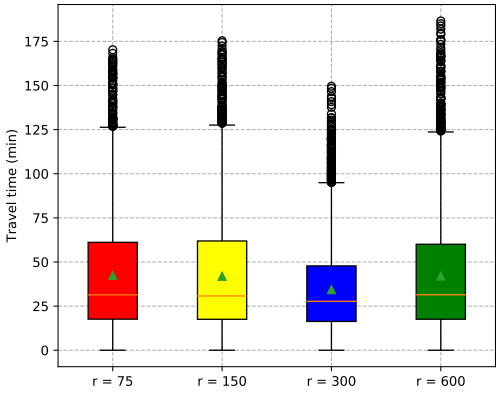
4.6.3 K-means Evaluation



(a) Transmitted messages routing step



(b) Accuracy of the traffic view



(c) Travel time

Figure 4.3: Results of the k-means evaluation.

With this evaluation, we want to verify the following questions: *(i)* how does the k-means algorithm impact SIC's performance? and *(ii)* What is the best value for k in the k-means algorithm for the evaluated scenario?

To define the number of k we used the Equation 4.6 varying the value of r, where $r \in \{75, 150, 300, 600\}$. In this way, to each value of r a different value of k is obtained. To each value of r, the following metrics were assessed: *(i)* transmitted messages routing step; *(ii)* accuracy of the traffic view; and *(iii)* travel time.

Figure 4.3 shows the results of the assessed metrics. In particular, Figure 4.3(a) shows the transmitted messages per routing step for each value of r. To understand the SIC

network's behavior, we need to remember Equation 4.6, which defines the number of k (i.e., sub-regions) based on the size of the communication range r. Thus, the smaller the communication radius (r), the greater the values of k. Consequently, the number of messages also increases because it is related to the number of sub-regions created by SIC (see Figure 4.3(a)). In this evaluation, we can identify two issues related to the value of r: *(i)* smaller values create an undesired overhead by increasing the number of messages disseminated throughout the network, and *(ii)* larger values can split the network connection, since key vehicles (i.e., the ones that might be responsible for creating knowledge about the traffic conditions of a specific region) may not participate in the dissemination procedure. The best performance is $r = 300$, which reduces the number of messages transmitted by approximately 30% when compared to the other values of r.

Figure 4.3(b) shows that both issues lead to the same problem, which is the degradation of the accuracy of the traffic view. On the one hand, the overhead created by small values of r potentially increases the packets' collisions and reduces the shared messages' delivery ratio. Large values of r potentially create gaps (areas in which vehicles do not know about the traffic condition) in vehicles' traffic view. As it can be seen, for $r = 600$, 60% of the vehicles have an accuracy lower than 65% in the traffic view, while $r = 300$ has an accuracy higher than 80% for 60% of the vehicles. The problems related to the overhead created by $r = 75$ can be seen when comparing with $r = 300$, in which it reduces the accuracy of the traffic view by approximately 15%.

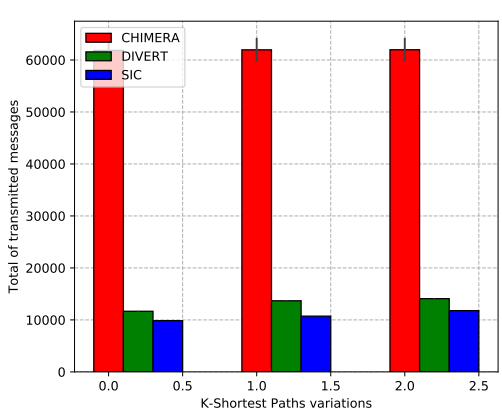
The effects of having better knowledge about the traffic conditions can be seen in Figure 4.3(c). The better traffic view provided by $r = 300$ paves the way to SIC compute alternative routes more efficiently. Thus, it reduces the travel time by approximately 40% when compared to the other values of r.

With this analysis, we can conclude that the best value for r is equal to the size of the communication range of the vehicles, which means that with this configuration, vehicles can have better sensing of the urban environment because the size of the sub-region is equivalent to their sensing capabilities. Besides, it does not produce an undesired overhead for the system and does not degrade the traffic conditions' knowledge.

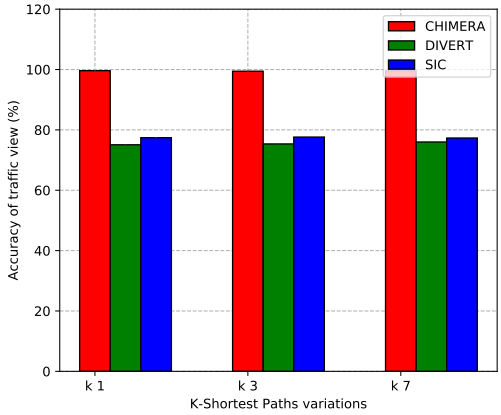
4.6.4 K-Shortest Paths Evaluation

In this evaluation, we want to assess: *(i)* how does the hyper-parameter k of the re-routing algorithm (i.e., k-shortest paths) impact the system scalability? and *(ii)* does the re-routing algorithm produce an undesired overhead? To do so, we define the following metrics: *(i)* total number of transmitted messages; *(ii)* accuracy of the traffic view; and *(iii)* CPU time.

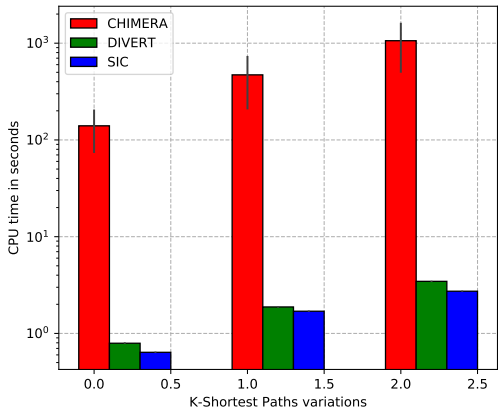
For this analysis, we used the CHIMERA and DIVERT solutions based on the k-shortest path algorithm compared with SIC. Figure 4.4 shows the results of the assessed metrics. In particular, Figure 4.4(a) shows the total number of transmitted messages. As it can be seen, the hyper-parameter k provides a slight increase in the number of transmitted messages. However, it does not degrade system efficiency for all solutions since they keep the same accuracy of the traffic knowledge (see Figure 4.4(b)). In this way, the increase in the number of transmitted messages is because some vehicles potentially



(a) Total number of transmitted messages



(b) Accuracy of the traffic view



(c) CPU time

Figure 4.4: K-shortest paths evaluation.

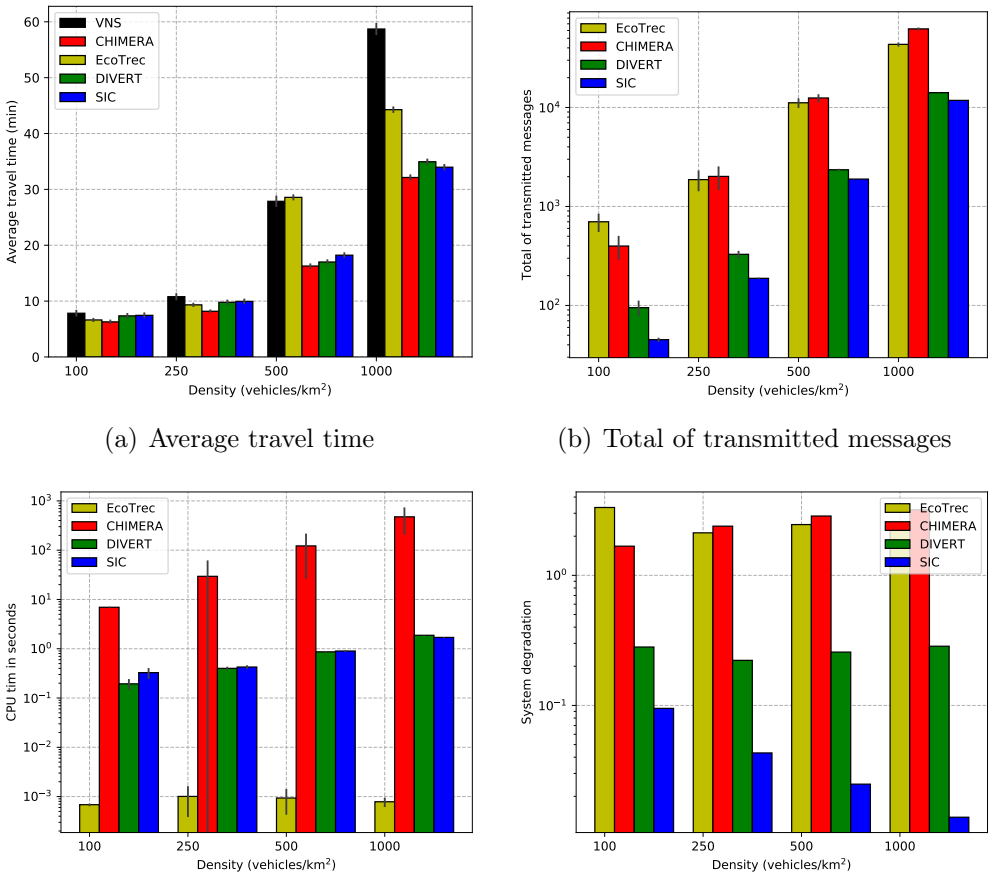
travel longer distances depending on the value of k. Therefore, vehicles tend to stay longer in the simulation (i.e., traveling in the scenario).

As the value of k increases, the CPU time also increases (see Figure 4.4(c)), which results from the number of paths that need to be computed. In this way, the CPU time has a higher impact on centralized solutions than distributed ones. Thus, CHIMERA presents a higher increase in the CPU time as the value of k grows, while DIVERT and SIC have only a slight increase in the CPU time. These results show the limitations related to scalability presented in CHIMERA and show that despite the increase in the CPU time, it does not decrease the system efficiency of DIVERT and CHIMERA, which are distributed solutions.

4.6.5 System Efficiency Evaluation

In this evaluation we assess SIC's efficiency compared to a traditional vehicular navigation system (VNS), CHIMERA, EcoTrec, and DIVERT. The following metrics were analyzed: *(i)* average travel time; *(ii)* total of transmitted messages; *(iii)* CPU time; and *(iv)* System degradation.

Figure 4.5 shows the results for the assessed metrics. Specifically, Figure 4.5(a) shows



(a) Average travel time

(b) Total of transmitted messages

(c) CPU time

(d) System degradation

Figure 4.5: System efficiency evaluation.

the average travel time in minutes as a function of the density. As it can be seen, VNS has the highest travel time for all densities, which is a result of the lack of an efficient re-routing algorithm. Hence, the vehicles are re-routed once they start their trip. On the other hand, the other ITS-based solutions provide an improvement in the travel time of at least 15 minutes compared to VNS. However, EcoTrec potentially produces some congestion due to its deterministic approach to re-route vehicles. CHIMERA, DIVERT, and SIC are non-deterministic solutions and overcome the limitation presented by EcoTrec (e.g., routing many vehicles through the same path, which potentially creates different congestion spots), consequently decreasing the travel time by approximately 25% in comparison to EcoTrec.

CHIMERA, DIVERT, and SIC have equivalent traffic efficiency, which is a consequence of the non-deterministic re-routing algorithm employed by them. The 10% improvement in the travel time of CHIMERA in respect to DIVERT and SIC is the result of its centralized architecture that provides a more accurate traffic view and the knowledge about all alternative routes taken by all re-routed vehicles (see Figure 4.4(b)). Thus, CHIMERA provides a slightly better traffic balance. However, this improvement seems to be costly when analyzing the overhead of the system, which increases the number of transmitted messages by 430% and 520% when compared to DIVERT and SIC, respectively (see Figure 4.5(b)). SIC has the lowest overhead, which is the consequence of its efficient traffic

reporting algorithm, which reduces the transmitted messages by 80%, 72%, and 16% compared to CHIMERA, EcoTrec, and DIVERT, respectively. In this context, even with its limited knowledge about the alternative routes taken by the other vehicles (since SIC vehicles do not know the route taken by all vehicles), SIC still can deal with vehicular mobility properly. A result of its efficient traffic reporting mechanism, which provides accurate traffic knowledge to improve mobility.

Centralized solutions tend to provide higher overhead because all vehicles need to report their traffic information to some RSU. The central server needs to compute the new routes to re-route the vehicles. Such an issue limits the system scalability by overloading the network and introducing an undesired delay for the system to compute alternative routes. Figure 4.5(c) shows the average CPU time over of the density of vehicles. As expected, for CHIMERA, the higher the density is, the higher is the CPU time.

Nevertheless, EcoTrec, DIVERT, and SIC have an offloading mechanism to avoid the computation burden on the central server, thus, dramatically reducing the CPU time. EcoTrec has the lowest CPU time because EcoTrec implements the shortest path algorithm and does not provide any mechanism to balance the traffic flow. Despite the better CPU time, it decreases the traffic efficiency creating different congestion spots (see Figure 4.5(a)). DIVERT, and SIC have slightly higher CPU time than EcoTrec, which results from the mechanism to balance the traffic flow implemented by them. However, DIVERT and CHIMERA have a CPU time lower than 2 seconds, which is 99% lower than the CPU time of CHIMERA. Such a result shows the efficiency of the offloading and re-routing mechanisms, which are essential for real-time re-routing.

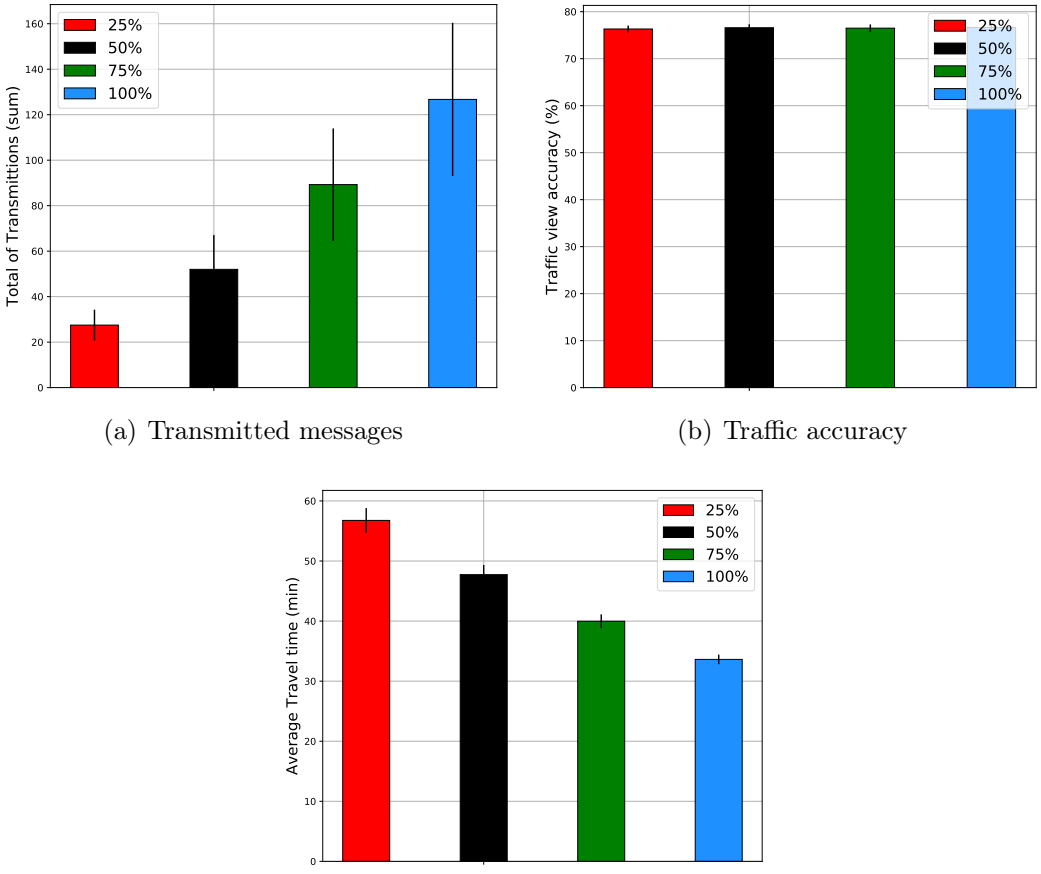
We assume that the shortest path implementation uses a Fibonacci heap for the complexity analysis. Thus, it has a time complexity of $\mathcal{O}(E + V \log V)$. EcoTrec offloads the re-routing computation in each vehicle and implements the shortest path algorithm; hence, it has a complexity of $\mathcal{O}(E + V \log V)$. On the other hand, CHIMERA, DIVERT, and SIC are based on the K-Shortest Path algorithm [68], which makes K calls to the shortest path algorithm, providing a complexity of $\mathcal{O}(K(E + V \log V))$. However, CHIMERA performs such algorithm for the set of vehicles that need to be re-routed (e.g., N_r), the overall time complexity of the system is $\mathcal{O}(N_r(K(E + V \log V)))$. For DIVERT and SIC, instead of computing the alternative routes for all vehicles, each vehicle computes the alternative routes only for itself. Before computing a new route, each vehicle needs to wait for *rank* milliseconds to receive the alternative routes of its vicinity, which is based on the distance to the traffic jam. Thus, the complexity of SIC is $\mathcal{O}(K(E+V \log V)+t_{rank})$. It is important to notice that the small difference in CPU time of DIVERT and SIC is a consequence of the methods used to establish the *rank* delay. Thus, we can conclude that they have equivalent complexity.

The overall efficiency of SIC can be seen by analyzing the system degradation metric in Figure 4.5(d). This metric summarizes the system efficiency by combining the network overhead, the knowledge about the traffic conditions, and the system scalability. Thus, the higher the degradation is, the lower is the efficiency of these metrics. As it can be seen, SIC has the lowest degradation, and as the density increases, the system degradation also decreases. In particular, SIC reduces the system degradation in 99%, 99%, and 96% in comparison to CHIMERA, EcoTrec, and DIVERT, respectively.

Considering these results, we highlight (*i*) each vehicle only needs to know the route taken by the vehicles in its vicinity to achieve a good traffic balance, since the global information about all the vehicle routes brings minimal benefits; (*ii*) SIC presents an increase of 5% regarding the travel time compared to CHIMERA, but with lower CPU time, transmitted messages and system degradation, providing lower overhead and complexity with higher system scalability; and (*iii*) the efficient performance of offloading the re-routing computation in each vehicle establish the way to an efficient real-time traffic re-routing system.

4.6.6 SIC Evaluation

This subsection evaluates SIC performance in terms of: *(i)* compliance ratio; *(ii)* re-routing frequency; *(iii)* route similarity; and *(iv)* data compression.



(a) Transmitted messages

(b) Traffic accuracy

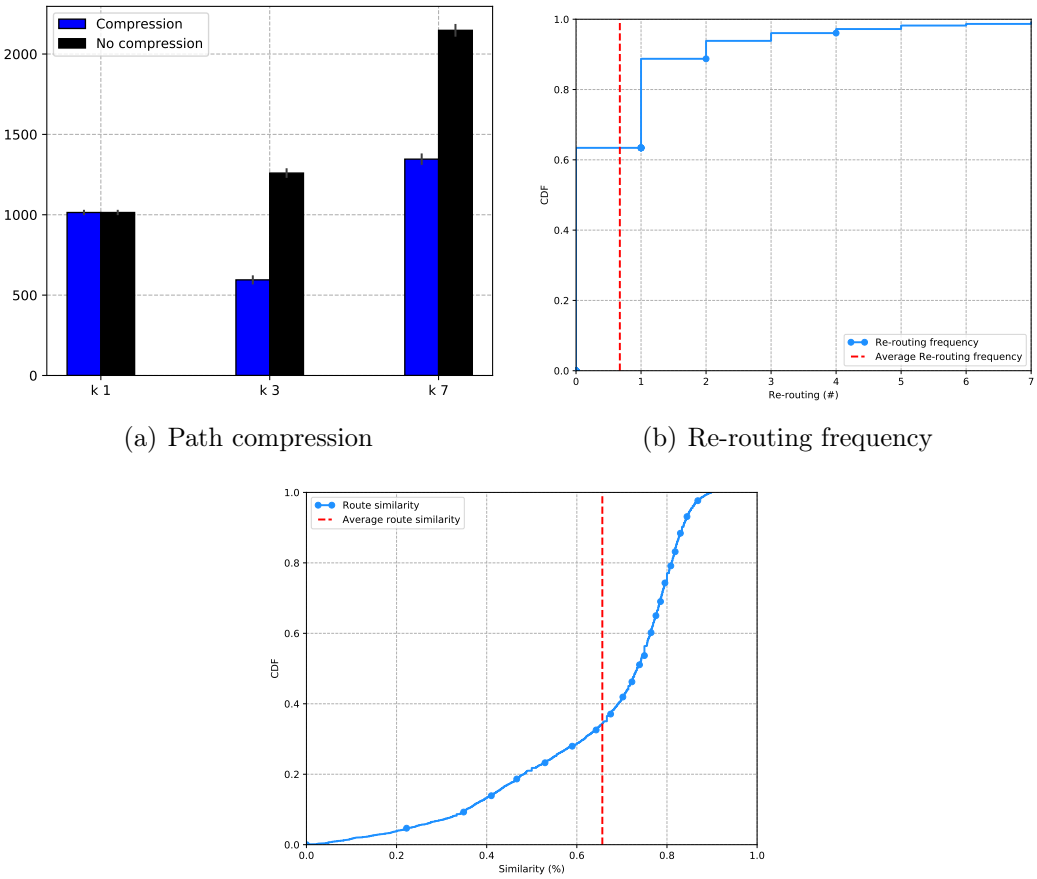
(c) Travel time

Figure 4.6: Penetration rate.

Figure 4.6 shows the performance of SIC in a scenario where only a percentage of vehicles use the system. Thus, we evaluate SIC in 25%, 50%, 75%, and 100% of the vehicles (i.e., penetration rate). As expected, the lower the number of vehicles using SIC is, the fewer messages are transmitted (see Figure 4.6(a)). However, even for a penetration rate of only 25%, SIC still provides good traffic estimation, which is higher than 75%, since a traffic report from a single vehicle in a road is enough to represent the traffic

condition in that road (see Figure 4.6(b)). In this way, SIC can compute reliable routes for the vehicles to improve their mobility. Nevertheless, as the penetration rate grows, more vehicles are re-routed, consequently improving traffic efficiency. It is worth noticing that the error in the traffic estimation of the roads that are not accurately estimated is lower than 10%, which means that SIC does not guide vehicles through congested roads assuming that they are free.

In summary, the penetration results have shown that SIC is a suitable solution for dealing with traffic mobility not only when some drivers do not share the traffic report but also when they do not follow the system's routes.



(a) Path compression

(b) Re-routing frequency

(c) Route similarity

Figure 4.7: SIC evaluation.

In SIC, the vehicles need to share alternative routes to enable cooperative traffic management. However, this approach may produce network contention and decrease the overall system performance if not handled properly, since the size of the message shared by the vehicles depends on the size of the scenario and also on the parameter K of the K-Shortest Paths algorithm [68]. Therefore, Figure 4.7(a) shows the results of the data compression mechanism implemented by SIC to reduce the size of the messages shared with the set of alternative routes. As it can be seen, SIC can reduce the size of the transmitted messages by up to 35% considering $K = 7$, which is a consequence of the effectiveness of the compression mechanism that uses the most overlapping paths to compress the others. It is important to notice that reducing the size of the message also

reduces the number of transmitted messages, consequently decreasing the overhead of the system.

Figures 4.7(b) and 4.7(c) show the results of the re-routing frequency and route similarity metrics as a CDF, respectively. As can be seen, most vehicles are not re-routed, and only 10% of the vehicles are re-routed more than once. These results show that SIC does not decrease the driving experience with frequent route changes to achieve its performance (see Figure 4.7(b)). Also, vehicles re-routed more than once, the traveled route is at most 30% similar to the planned route (i.e., the route planned at the beginning of the simulation). This shows that multiple route changes are only performed when indispensable to improving mobility. Thus, SIC does not recommend routes to vehicles that were previously planned to themselves. Finally, it is worth noticing that most vehicles are not re-routed because it is not needed (i.e., they do not pass through congested areas). The average route similarity is close to 50% because the many vehicles still travel along some part of their planned route before being re-routed, and also more than 50% of the vehicles do not change their routes at all.

4.7 Chapter Conclusions

This chapter introduced SIC, a cooperative routing algorithm to improve traffic efficiency. SIC was designed based on two major principles for vehicular traffic management: *(i)* real-time vehicular traffic re-routing; and *(ii)* network contention minimization. SIC offloads the route computation in each vehicle, reducing the computation time and the communication burden on the server, consequently providing better scalability to the system. Furthermore, it employs a cooperative re-routing algorithm in which the vehicles are aware of their neighbors' routes, thus providing better traffic management. The results have shown that SIC provides a suitable architecture for traffic re-routing, which produces a low overhead and low complexity and CPU time (which enables a real-time system), consequently enabling a highly scalable system cooperative re-routing algorithm.

However, SIC only considers a single-objective to re-route vehicles (i.e., the traffic condition), which is far from the desired requirements for future TMSs, because several different urban aspects can be taken into consideration during route planning decisions, such as distance, fuel consumption, CO_2 emissions, scenery, and even safety risks. All vehicles are re-routed according to the same criteria, but different users may have different preferences according to their path planning decisions. For instance, cautious users may prefer a safer but longer route, while users with limited time tend to prefer faster routes [15]. Therefore, this thesis will address the multi-objective and personalized re-routing issues for traffic management systems considering different urban aspects and their spatiotemporal correlation in the next chapter.

Chapter 5

Multi-objective Vehicle Re-Routing Based on Spatiotemporal Information

Research question 2: *How to enable efficient and personalized multi-objective re-routing without creating different congestion spots?*

This chapter introduces Safe and Sound (SNS), a non-deterministic multi-objective vehicular traffic re-routing system that enables personalized route planning. As a use case, we have considered traffic conditions and public safety issues as urban aspects to perform multi-objective re-routing. However, the proposed solution can naturally work with other urban aspects such as fuel consumption, road pavement, CO_2 emissions, scenery, city view, etc.

To deal with network issues and enable real-time route planning, we used the architecture proposed in Chapter 4. In this way, some tweaks were necessary for SNS to reach the desired performance regarding the personalized multi-objective re-routing algorithm. To avoid redundant discussion, the main focus of this chapter is the personalized multi-objective re-routing algorithm. Thus, only a brief description of the tweaks incorporated into the architecture will be discussed.

By the end of this chapter, we will have provided solutions to reduce the following issues: *(i)* how to provide efficient multi-objective re-routing that avoids the problem of creating different congestion spots; and *(ii)* how to enable personalized re-routing, in which vehicles can decide which urban aspects are more relevant to them during the route planning.

The rest of this chapter is organized as follows. Section 5.1 presents the overview of the proposed system. Section 5.2 introduces the mechanisms used for discovering risky areas and for exploring spatiotemporal information. Section 5.3 describes the solutions for multi-objective re-routing, while Section 5.4 describes the proposed non-deterministic multi-objective re-routing algorithm for improving traffic efficiency and the safety of driver and passengers. Finally, Section 5.5 presents the performance analysis for the algorithms proposed by SNS and Section 5.6 concludes the chapter.

5.1 SNS Overview

SNS uses the solutions proposed in Chapter 4 as underline architecture to meet the network requirements of an efficient TMS and deal with scalability issues and enable real-time re-routing. However, besides the knowledge about traffic conditions, the system needs to know about public safety issues and perform multi-objective re-routing based on traffic mobility and safety risks. In this way, sources of safety-related data were incorporated in the SNS architecture to provide such information. The overall architecture of the system is shown in Figure 5.1.

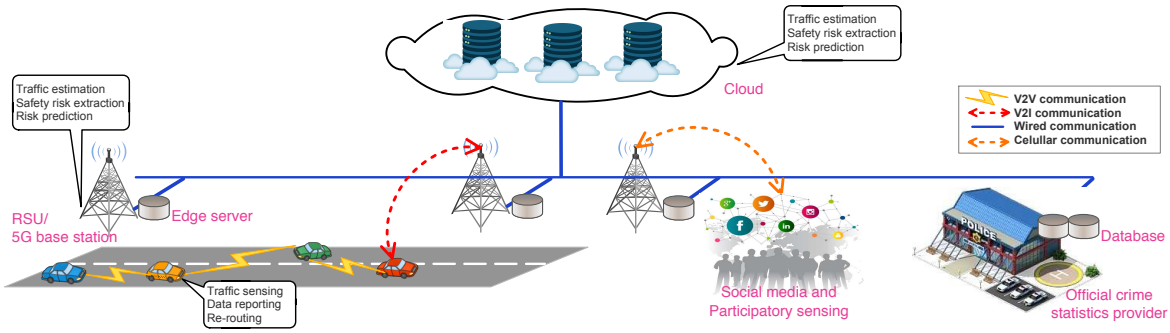


Figure 5.1: The architecture employed by SNS, presenting the main components that compose it and their activities.

For the sake of description, the SNS architecture is composed of vehicles, edge servers, criminal-related data providers, and the cloud. Vehicles with on-board units (OBU) can communicate with the roadside infrastructures (e.g., RSUs, 5G base station, etc.), edge servers, and with the remote cloud over either 5G or vehicular networking to report traffic-related data, to request/receive urban-related dynamics (e.g., traffic conditions and risky areas), and to re-routing themselves for improving their mobility and security. On the other hand, edge servers widely deployed at roadside infrastructures can provide processing and storage resources on the network edge, enabling fast responsiveness. In this sense, each edge server is responsible for building local knowledge about traffic conditions in its coverage based on vehicles report and estimating risky areas based on public safety information. Naturally, the cloud is responsible for building global knowledge about traffic conditions and global awareness about future risk areas to provide the vehicles an understanding beyond the edge servers' experience and deal with resource constraints presented by them. Finally, the criminal-related data providers are responsible for feeding the system with criminal reports to build pieces of knowledge about city-wide illegal activities. Those providers can be: *(i)* police departments providing the history of official crime statistics; and *(ii)* people providing a more dynamic knowledge and real-time sensing based on crowdsourcing approaches (e.g., participatory sensing applications and social media), which is not possible by using just historical data.

In this scenario, vehicles can periodically re-route themselves to improve the overall traffic efficiency while decreasing safety risks according to their preferences. Vehicles need to sense the urban environment and provide traffic-related information to the servers using an efficient traffic-aware data reporting mechanism detailed described in Chapter 4.

On the other hand, edge servers need to explore the spatial and temporal correlation of different criminal activities in Section 5.2. Therefore, with both pieces of knowledge (e.g., traffic conditions and safety dynamics), vehicles can employ an efficient personalized and cooperative context-aware traffic re-routing algorithm described in Section 5.4 to improve their mobility while avoiding their chosen safety risks. To explain each procedure employed by the system, we defined the following scenario modeling:

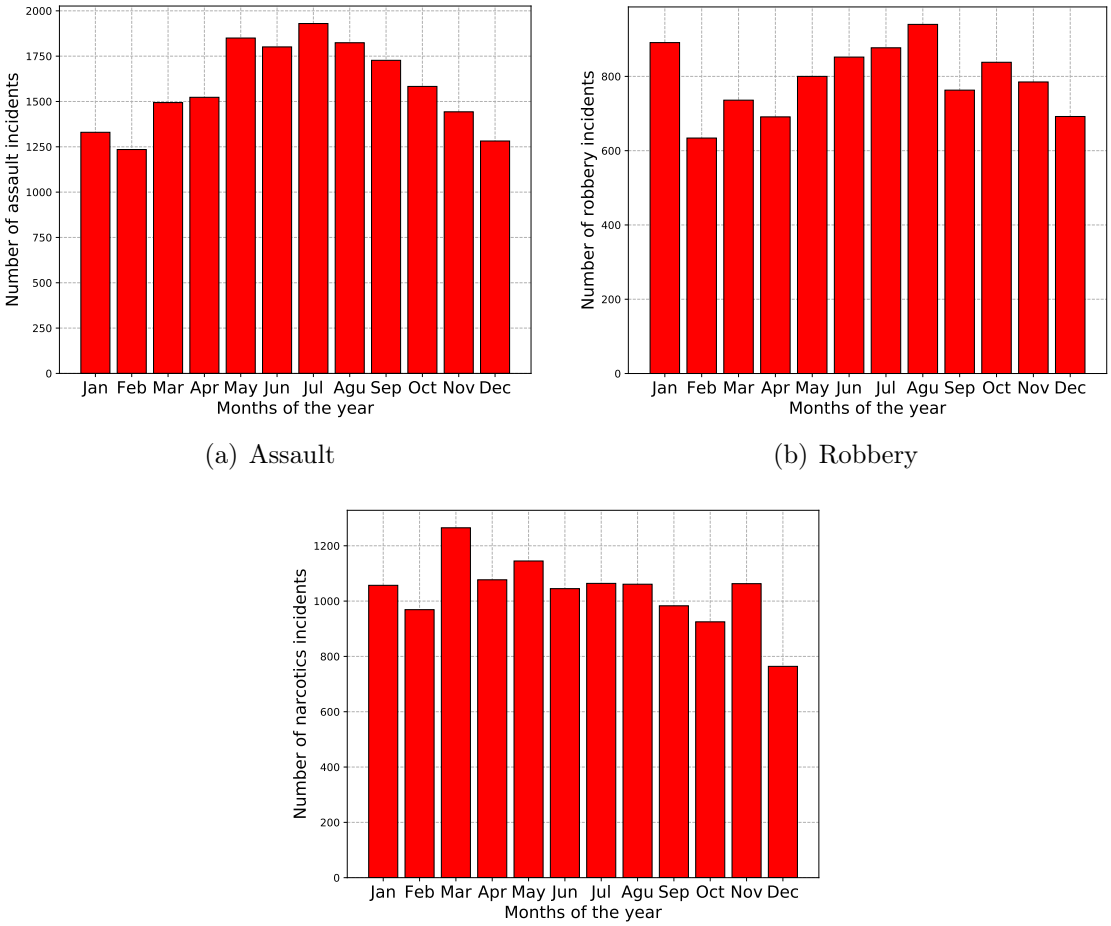
Urban scenario modeling: *Considering the road network represented by a direct graph $G = (V, E)$, in which the set of vertices V represents the scenario intersections, while the set $E \subseteq V \times V$ corresponds to the road segments, e.g., the road segment $uv \in E$ represents the road segment connecting the intersections u and v. Each road segment $uv \in E$ has a length represented by l_{uv} and a shape S_{uv}, which is a set of points $p \in S_{uv}$ defining its real geographic shape. Moreover, each road segment also has two attributes τ_{uv} and r_{uv}, defining its current traffic condition and its safety risk, respectively. Each vehicle in the road network is represented by the set N, in which each vehicle $n \in N$ has a pair of origin $s \in V$ and destination $t \in V$, such that $s \neq t$, and is associated to a path $P \subseteq E$ connecting s to t, which defines the vehicles' route. Eventually, the traffic condition and the safety risk of a path P is defined as $\tau_P = \sum_{uv \in P} \tau_{uv}$ and $r_P = \sum_{uv \in P} r_{uv}$.*

5.2 Discovering Risky Areas and Exploring Spatiotemporal Information

To provide a real-world scenario for SNS, the characteristics of a real city and its data about criminal activities were used by the system. In this way, Chicago is a city where both pieces of information are publicly available. The city characteristics are available from the *OpenStreetMaps* tool. Simultaneously, the public safety dataset is available by the Open Data Chicago (ODC), which provides well-structured data about criminal activities in the city. A set of features are provided by each criminal report, including a brief description, timestamp, type of crime, place, geographic coordinates (e.g., latitude and longitude). It is worth noticing that the pre-processing methods applied for extracting and classifying text from criminal-reports and social media posts to identify the type of crime are beyond the scope of this thesis. Figure 5.2 shows the crime distribution in Chicago, for each month of the year of 2018 considering the three more frequent crimes: *(i)* assault; *(ii)* robbery; and *(iii)* narcotic-related crimes.

To capture the behavior of each criminal activity, we have to understand that each criminal event is motivated by different dynamics related to the environment [34]. Thus, the same region/neighborhood can provide different conditions depending on the day, time, month, weather condition, etc. Consequently, either increasing or decreasing the criminal opportunities within it. In other words, criminal activities have a spatiotemporal correlation, which produces hotspots (i.e., regions with a high number of crimes) for specific illegal activities along the day.

People are the primary targets of criminal activities, and they have daily routines [34]. This relation produces spatiotemporal patterns at the locations of dangerous areas. Dangerous areas are considered regions that are likely to have an elevated amount of crimes



(a) Assault

(b) Robbery

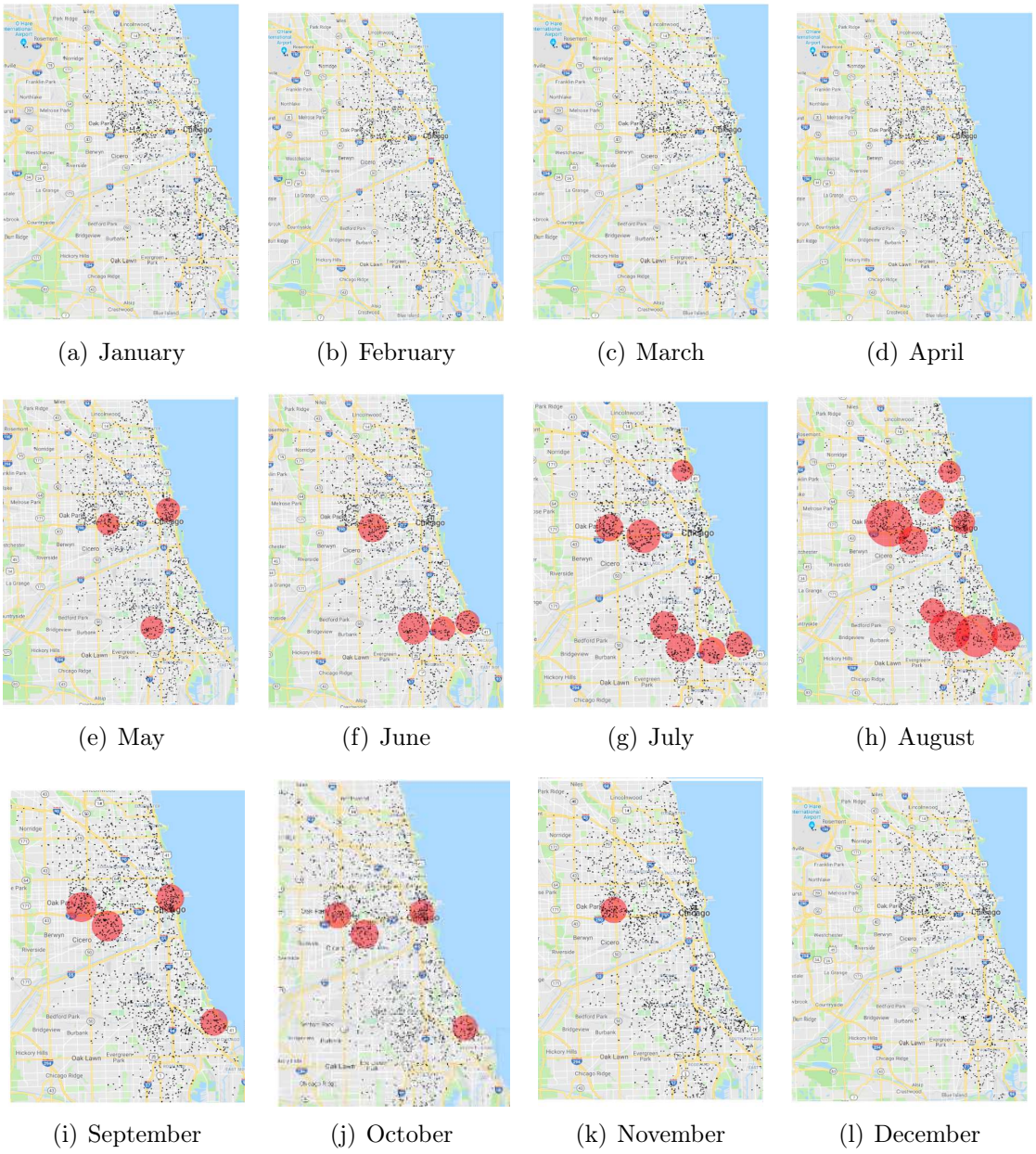
(c) Narcotics

Figure 5.2: Criminal incident distribution of each month during 2018 considering Assault, Robbery and Narcotics incidents.

according to the day and time. Figure 5.3 shows a density-based clustering for crimes that happened on Mondays afternoon (from 12:00 to 18:00) in each month of 2018 in Chicago. As it can be seen, during the winter, no cluster was created, which means that fewer crimes happened as a result of the reduced number of people on the streets, consequently decreasing the criminal opportunities. On the other hand, during the summer and vacation months, we can see many clusters resulting from the higher number of people on the streets, contributing to more criminal opportunities.

The clustering method provides a good representation of criminal density in the city. However, using this method is challenging to measure the criminal density in each road since all roads in the same cluster potentially will have the same measure. Also, the circle shape of the cluster might introduce false positives in the system because roads that are safe at a time t can potentially be included in the cluster as a consequence of its shape.

In this scenario, to overcome the issue above and also to identify the spatiotemporal correlation, SNS implements a mechanism to measure the spatial density of each criminal activity based on its geographic coordinates considering a predefined time window (t_{start} and t_{end}). Therefore, let C be a set of criminal incidents that had happened in the city, in which each crime $c \in C$ has its location (e.g., geographic coordinates), date-time, and type



(a) January (b) February (c) March (d) April

(e) May (f) June (g) July (h) August

(i) September (j) October (k) November (l) December

Figure 5.3: Density-based clustering of the crimes happened in Mondays during the afternoon (from 12:00 to 18:00) to each month of 2018 in Chicago.

(e.g., assault, robbery, narcotics, weapons violation, kidnapping, etc.). The spatial density is defined based on a Gaussian Kernel Density Estimation (KDE), in which estimates the criminal density at a point p based on the set C of criminal events related to a particular type of crime that happened between the time window t_{start} and t_{end}, given by $\theta(p)$

$$\theta(p) = \frac{1}{|C|} \cdot \sum_{c \in C} \frac{1}{h\sqrt{2\pi}} e^{\left(-\frac{1}{2}\left(\frac{||c-p||}{h}\right)^2\right)} \tag{5.1}$$

where $||c-p||$ is the Euclidean distance between the points c and p and h is the bandwidth used. The bandwidth h defines Gaussian kernel spread. It controls the smoothness of the estimated density, which was defined based on [59].

Figure 5.4 shows an example of the criminal density estimation for one week of October of the criminal dataset. Each density plot represents the estimations in the entire city according to the day of the week and period of the day (dawn, morning, afternoon, and night). To each period, the set of the criminal events were filtered by the time window (t_{start} and t_{end}) defined by that period. As it can be seen, this representation can capture the hotspots created by the criminal activities (see the dark colors in the plots). Besides, it overcomes the clustering methods' shape-related problem by fitting the estimations according to the density.

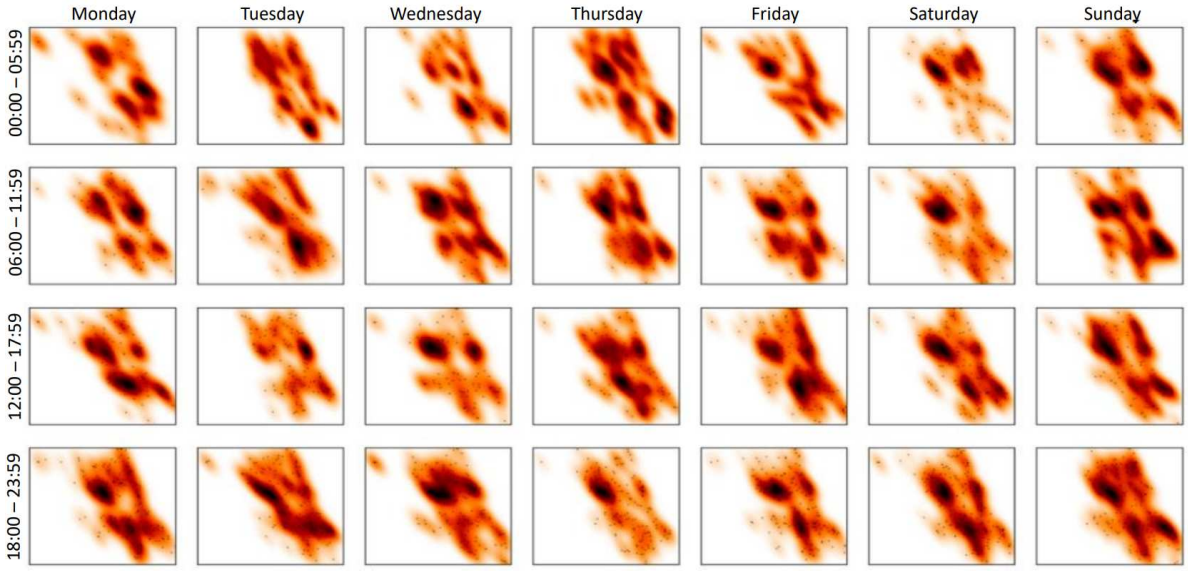


Figure 5.4: Criminal density estimation during one week of the criminal dataset considering four periods of the day: *(i)* dawn, from 00:00 to 05:59; *(ii)* morning, from 06:00 to 11:59; *(iii)* afternoon, from 12:00 to 17:59; and *(i)* night, from 18:00 to 23:59.

After estimating the criminal density over the entire city, those estimations need to be mapped to the roads to provide well-structured information to the system and to enable it to measure the risks over the vehicle's routes. In this way, the criminal density is mapped according to the shape of each road. In other words, the criminal density of the road $uv \in E$ is the average of estimations over the set of points p that defines the shape of the road uv, defined as follows:

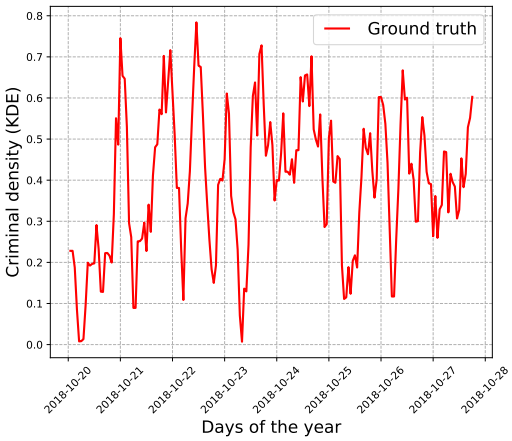
$$\sigma(uv) = \sum_{p \in S_{uv}} \theta(p), \tag{5.2}$$

where S_{uv} is the set of points p that define the shape of the road segment $uv \in E$. Then, the criminal activity of each road segment can be obtained as the normalized densities:

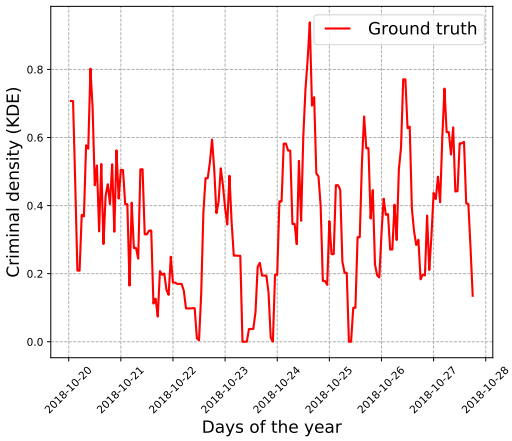
$$r_{uv} = \frac{\sigma(uv)}{\sum_{u'v' \in E} \sigma(u'v')}, \tag{5.3}$$

where r_{uv} is proportional to the probability of observing a crime incident at road $uv \in E$. Analogously, the criminal activity of a path P is r_P, which describes the total criminal density of the path, e.g., $r_P = \sum_{uv \in P} r_{uv}$.

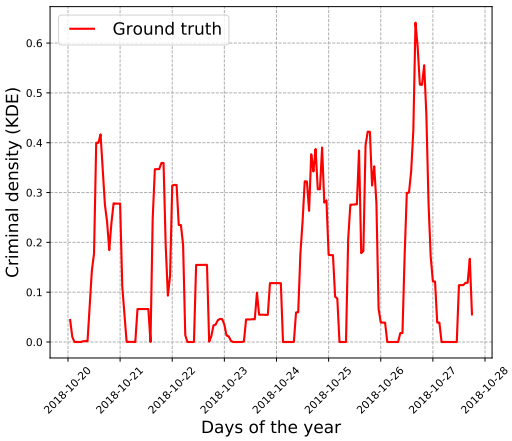
In this context, to explore the spatial and temporal correlation of each criminal activity along each day, we defined the time window size to be 1 hour and then compute the KDE for all roads considering the set of crimes in that period for each type of crime type. Hence, creating a time series of criminal density in each road considering each type of crime. Figure 5.5 shows an example of the temporal correlation for the three more frequent crimes in the dataset considering downtown Chicago during one week of October 2018.



(a) Assault



(b) Robbery



(c) Narcotics

Figure 5.5: Temporal correlation for the Chicago's downtown during one week of October of 2018.

With this spatiotemporal correlation, SNS can extract safety risks over the roads in the entire city based on the desired timestamp. This information, jointly with the traffic view created by the vehicular data sharing employed by the system (Chapter 4), are the building blocks for efficient multi-objective re-routing to improve not only the safety of drivers and passenger but also the overall traffic efficiency over the city.

5.3 Strategies for Multi-objective Vehicle Re-routing

The key concerns on multi-objective re-routing (e.g., mobility and safety risks) are: *(i)* the path with the highest traffic efficiency path is not necessarily the safest one and vice versa; and *(ii)* how to guide vehicles properly, balancing the traffic flow to avoid creating different congestion spots.

The first concern is a multi-criteria optimization problem, which can be solved by combining the two attributes of each road into a single one as a weighted-sum [49], then use the shortest path approach. We can define the solution for such problem formally as:

$$P = \arg\min_{P \in E} \left(\alpha \cdot \tau_P + (1 - \alpha) \cdot r_P\right), \tag{5.4}$$

where P is a (s-t)-path, and $\alpha \cdot \tau_P + (1 - \alpha) \cdot r_P$, is a weighted sum to measure both attributes into a single one.

Although this approach could be an option, it is not straightforward to define the value of α properly since it can depend on its application, and their values could be measured in different units. In this way, instead of combining both attributes into a single one, another approach is to model such a problem as an instance of the Resource-Constrained Shortest Path problem (RCSP) [39], which is NP-hard. The key idea is to find the fastest path whose risk does not exceed a threshold λ (e.g., the resource). Formally, considering a vehicle n with current position s and destination t, the problem to find the safest route consists in:

$$\min\{\tau_P \mid P \text{ is a (s-t)-path and } r_P \leq \lambda\} \tag{5.5}$$

Despite weighted-sum shortest path and RCSP being options for solving multi-objective vehicle re-routing, they are deterministic, which potentially leads to the second concern (e.g., these solutions potentially create congestion in other areas). To avoid this problem, two or more vehicles having similar origin s and destination t should receive different paths whenever possible. In this context, we propose a multi-objective vehicle re-routing approach based on Pareto-optimality to balance the traffic along the paths in the Pareto set.

5.4 Personalized Multi-objective re-routing based on Pareto set

The Pareto set $\mathcal{P}$ for a vehicle with origin s and destination t, is a set of paths that optimize both metrics mobility and safety risk (e.g., τ_P and r_P). Formally, let P_{old} be the current route of a vehicle (assumed to be the shortest path) and P_{new} be a path in the road network G linking s to t. Thus, the Pareto set will be composed of the set of paths that respect the following condition:

$$\mathcal{P} = \begin{cases} P & \text{if } \tau_{P_{\text{new}}} \leq \tau_{P_{\text{old}}} \text{ and } r_{P_{\text{new}}} \leq r_{P_{\text{old}}},\ \forall P_{\text{new}} \in G \\ \emptyset & \text{otherwise} \end{cases} \tag{5.6}$$

On computing the Pareto set, SNS defines the set of potential routes to re-route the vehicle. Thus, to avoid issues related to deterministic solutions, SNS applies a probabilistic approach to balance the traffic flow over the Pareto set's potential routes. The set of potential routes is determined based on a Pareto curve, which provides the fastest route (e.g., optimal) to every possible safety risk value of each route. Figure 5.6 shows an example of a Pareto curve to optimize both mobility and safety. In this example, the safer the route is, the slower it becomes.

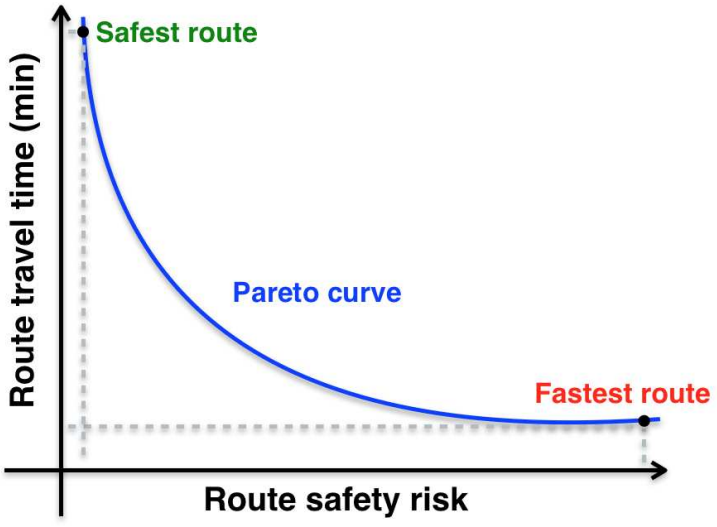


Figure 5.6: Example of a Pareto curve for mobility and safety

The safety risk of a path P is defined as $r_P = \sum_{uv \in P} r_{uv}$. Thus, considering $r_P = \lambda$ the safety risk of the new path must be a number between $[0, \lambda]$. In this scenario, the problem of finding the Pareto set of a path with origin s and destination t, is similar as finding the fastest path P with $r_P \leq \varphi$ such that $\varphi \in \{0.01, 0.02, \ldots, \lambda\}$, thus, satisfying the following recurrence:

$$T(v,\varphi) = \begin{cases} 0 & \text{if } \varphi = 0 \text{ and } v = s, \\ \infty & \text{if } \varphi = 0 \text{ and } v \neq s, \\ \min \begin{cases} T(v, \varphi - 0.01) \\ \min\limits_{u: r_{uv} \leq \varphi} \{T(u, r_{uv} - 0.01) + \tau_{uv}\} \end{cases} & \text{Otherwise} \end{cases}, \tag{5.7}$$

Such recurrence naturally derives a recursive algorithm with exponential time complexity. However, it can be solved using a dynamic programming approach with a complexity of $O(|E|\lambda)$, which has pseudo-polynomial complexity since, in our scenario, the value of λ is not arbitrarily large, since the safety risk of each road is at most 1.

Algorithm 4 describes the dynamic programming approach used for computing the Pareto set. In the algorithm, DP is a dynamic programming table used to memorize the traffic efficiency value to reach each vertex v from a vertex s with risk φ, for $s, v \in V$. Such path can be stored by table ψ, in which $\psi[v, \varphi]$ is a predecessor of v in the (s, v)-path with risk φ. The first and the second loops (lines 3-10) are responsible for preparing

Algorithm 4: Dynamic programming algorithm for computing the Pareto set.

Input : s // *current position of vehicle* n
t // *destination of vehicle* n
λ // *maximum risk of the path*
Output: Pareto set of paths stating in s and ending in t with risk at most λ.

```
1  DP ← [ ] ; // table to store T(v, φ)
2  ψ ← [ ] ; // stores predecessor vertex in the route DP[v, φ]
3  foreach φ ∈ {0.01, 0.02, ... λ} do
4  |   ψ[s, φ] ← ∅;
5  |   DP[s, φ] ← 0;
6  end
7  foreach v ∈ V \ {s} do
8  |   ψ[v, 0] ← ∅;
9  |   DP[v, 0] ← ∞;
10 end
11 foreach φ ∈ {1, 2, ..., λ} do
12 |   foreach v ∈ V do
13 |   |   ψ[v, φ] ← ψ[v, φ − 0.01];
14 |   |   DP[v, φ] ← DP[v, φ − 0.01];
15 |   |   foreach uv ∈ E do
16 |   |   |   if r_uv ≤ φ then
17 |   |   |   |   if [v, φ] > DP[u, r_uv − 0.01] + τ_uv then
18 |   |   |   |   |   ψ[v, φ] ← u;
19 |   |   |   |   |   DP[v, φ] ← DP[u, r_uv − 0.01] + τ_uv;
20 |   |   |   |   end
21 |   |   |   end
22 |   |   end
23 |   end
24 end
25 return DP[t, ], ψ;
```

the DP and ψ tables in the base cases of the recurrence, which mark empty paths by clearing the predecessor of source s, and mark (s, v)-paths as unfeasible whenever the demanded risk is $\varphi = 0$. Later, the algorithm fills up DP table, for the following risk values $\varphi \in \{0.01, 0.02, \ldots, \lambda\}$: the best traffic efficiency is either the same as that of risk $\varphi - 0.01$ (line 14), or is achieved by following some non empty (s, v)-path from s to some predecessor u, and edge uv (lines 15-22). Possible predecessors are those vertices u for which $r_{uv} \leq \varphi$, and the corresponding (s, v)-path in this case has traffic efficiency $DP[u, r_{uv} - 0.01] + \tau_{uv}$ (line 19). Finally, the Pareto set can be obtained through line t for all values of φ.

Figure 5.7 shows a digraph $G = (V, E)$ with edges having two weights in the form τ_{uv}/r_{uv} representing traffic efficiency and safety risk, respectively, in a certain scenario. Table 5.1 exhibits the values obtained for DP when we run Algorithm 4. In this example scenario, if a vehicle has origin s and destination t, with safety risk of the shortest path of $\lambda = 10$. It is important to stress that the Pareto set for the scenario mentioned above is $DP[t,]$ (e.g., the whole line).

Each vehicle needs to compute its route according to its preference (e.g., based on which crime each driver wants to avoid) to enable personalized safety-based re-routing. Therefore, SNS offloads route computation from the cloud and edge server to each vehicle, which also improves system scalability as shown in Chapter 4. To avoid replicated descriptions, the mechanisms to offload the re-route computation, enable the cooperative re-routing, and provide data compression will be skipped in this chapter since they are based on the solutions presented in the previous chapter. Thus, please, refer to Chapter4 for a detailed description of these mechanisms.

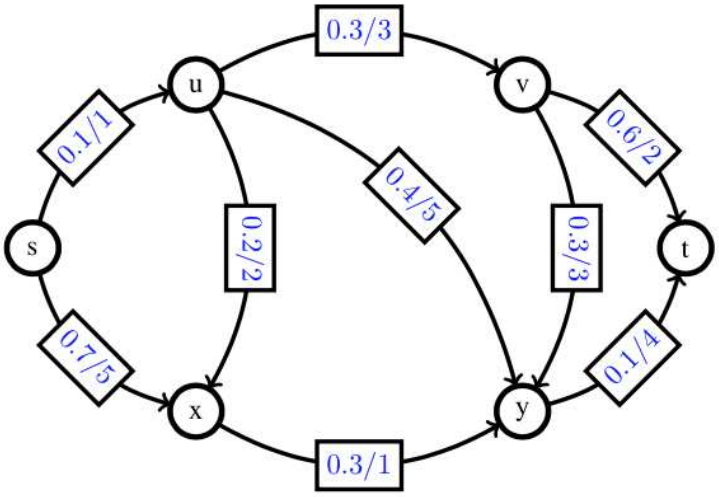


Figure 5.7: Digraph $G = (V, E)$ with traffic condition τ_{uv} and safety risk r_{uv} representation built by SNS.

Table 5.1: Dynamic Programming table for the Digraph of Figure 5.7

$v \backslash \varphi$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
s	0	0	0	0	0	0	0	0	0	0	0
u	∞	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
x	∞	∞	∞	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
v	∞	∞	∞	∞	0.4	0.4	0.4	0.4	0.4	0.4	0.4
y	∞	∞	∞	∞	0.6	0.6	0.5	0.5	0.5	0.5	0.5
t	∞	∞	∞	∞	∞	∞	1.0	1.0	0.7	0.7	0.6

Therefore, the entire system works as follows. The server will alert the vehicles with updated traffic view and with estimated safety risks on detecting signs of congestion. This process triggers the re-routing, which is composed of two main phases: (*i*) knowledge dissemination; and (*ii*) cooperative route computation. The first phase provides the updated pieces of knowledge about traffic conditions and safety risks to the vehicles that potentially will be affected by the congestion, while the other one is responsible for optimizing the re-routing algorithm to achieve better traffic management.

A popularity measure is associated with each road segment to avoid creating new congestion spots during multi-objective re-routing. A new congestion spot might come up if many vehicles take the same road segment within the same future time window. Hence, we define the demand d_{uv} of a road segment $uv \in E$, as the number of vehicles that have the road uv included into their potential paths of the Pareto set. This information can be obtained after the cooperative route sharing process.

Thus, by using the entropy of information theory, each vehicle can compute the entropy of each path P of the Pareto set $\mathcal{P}$ received from its neighborhood, given by the function $\xi(P)$:

$$\xi(P) = -\sum_{uv \in P} \frac{(r)^t \cdot d_{uv}}{r_{uv} \sum_{uv \in E} d_{uv}} \log \frac{d_{uv}}{\sum_{uv \in E} d_{uv}}, \tag{5.8}$$

where $(r)^t$ is the average road risk estimation in time t, r_{uv} and d_{uv} is the risk and the demand of the road segment $uv \in E$, respectively.

Moreover, based on the entropy of each path, each vehicle estimates the popularity of the path based on $e^{\xi(P)}$. Thus, focusing on providing a better distribution of the traffic flow, the route selected for each vehicle is given by the least popular path of the Pareto set. In other words, P is defined as:

$$P = \arg\min_{P \in \mathcal{P}} \left(e^{\xi(P)} \right), \tag{5.9}$$

where P is an (s-t)-path and $\mathcal{P}$ is the set of paths that compose the Pareto set.

5.5 Performance Analysis

This section analyzes the performance of SNS concerning its architecture, network performance, personalized traffic re-routing with spatiotemporal correlation, and criminal density prediction. Subsection 5.5.1 introduces the simulation platform, describing the tools, scenario, and analyzing metrics. Subsection 5.5.2 compares SNS with a centralized architecture to deliver the same service name as EBPOP [16] regarding its network mobility and safety performance. Subsection 5.5.3 compares SNS with literature solutions for vehicular traffic re-routing. At last, Subsection 5.5.4 shows the results of the personalized re-routing algorithm.

5.5.1 Methodology

The simulation platform is composed of the simulator of urban mobility, SUMO [7] version 0.30.0, the network simulator OMNeT++ [63] version 5.0 and also the vehicular networking framework Veins [61] version 4.6. The road network is composed of a fragment of 50 km^2 from Chicago, obtained using OpenStreetMap. The traffic mobility was produced using the TrafficModeler [55] tool to ensure realistic traffic mobility, resulting in a total of five thousand routes. The number of routes was defined to create heavy traffic and congestion (i.e., the travel time for a majority of the cars is significantly higher than the free-flow travel time). In this way, we generate the traffic at a constant rate by deploying one car each second in the simulator from one side of the scenario to the other one. By default, the shortest travel time paths are automatically calculated and assigned to each vehicle at the beginning of the simulation based on the road speed limits. The presented results have a confidence interval of 95%, and Table 5.2 shows additional parameters used in the simulation.

5.5.2 Network Overhead, Traffic Management and Safety Risk Analysis

With this analysis, we evaluate the SNS performance in terms of network cost, system scalability, complexity time, traffic efficiency, and safety risk compared to EBPOP [16] (e.g., a centralized version of SNS). In this way, the following metrics were assessed:

Table 5.2: Simulation parameters

Parameters	**Values**
Channel frequency	5.890e9 Hz
Propagation model	*Two ray*
Transmission power	2.2 mW
Communication range	300 m
Bit rate	18 Mbit/s
PHY model	IEEE 801.11p
MAC model	EDCA
Max hop count	10
Scenario	Chicago
Scenario size	50 km^2
Re-routing interval	450 s

- **Transmitted messages**: the total number of transmitted messages in the system to deliver its service per routing step. This metric includes traffic reporting transmissions and route recommendation, and traffic view dissemination. A high number of messages transmitted is a strong indication of redundant and unnecessary transmissions.

- **Traffic estimation accuracy**: the percentage of roads with estimated traffic conditions equal to the real one (extracted from the simulator) at each re-routing interval. Low accuracy means that either the proposed traffic reporting is not a suitable mechanism for representing the traffic conditions on the roads or the multi-hop data dissemination protocol is not able to deliver the traffic view to the target vehicles (e.g., vehicles that need to receive the traffic view to improving their mobility).

- **CPU time**: is the time spent by the re-routing algorithm to compute the new alternative route to the intended vehicles. A longer CPU time potentially introduces latency to the system, consequently degrading its overall performance.

- **Relative travel time**: the ratio between the vehicle's travel time using one of the re-routing approaches and its travel time without any re-routing solution at all. This metric summarizes the travel time reduction for each vehicle.

- **Relative safety risk**: the ratio between the safety risk of the vehicle's route using one of the re-routing approaches and its route safety risk without any re-routing solution at all. This metric summarizes the safety risk minimization for each vehicle.

Figures 5.8, 5.9 and 5.10 show the results for network costs, system scalability and time complexity metrics for SNS and EBPOP. In particular, Figure 5.8 shows the transmitted messages as a function of the simulation time according to each re-routing step. As expected, EBPOP transmits substantially more messages than SNS since it does not implement any mechanism to reduce traffic reports. Thus, each vehicle needs to report its traffic measurements to the server periodically (e.g., each re-routing step). Moreover, due

to the centralized approach implemented by EBPOP, whenever a vehicle passes through congestion (e.g., identified by the server), the server/fog needs to send a message with the already computed alternative route to such vehicle using the closest RSU. Hence, such a fact further increases the number of transmissions and reduces system scalability because this procedure potentially introduces undesired latency to the system in high-density scenarios.

On the other hand, due to SNS's efficient traffic reporting mechanism, it can reduce the number of transmissions to deliver its service. Compared to EBPOP, SNS reduces the number of transmissions by about 85%, on average. Besides, like EBPOP, SNS increases the number of transmissions according to the density of vehicles. However, such an increase is much lower when compared to the increase presented by EBPOP, which means that SNS provides better system scalability. In particular, during the peak density in our simulations (see values with simulation time between 1 and 2 hours in Figure 5.8), EBPOP transmits about 2500 messages to deliver its service, while SNS transmits less than 400 messages.

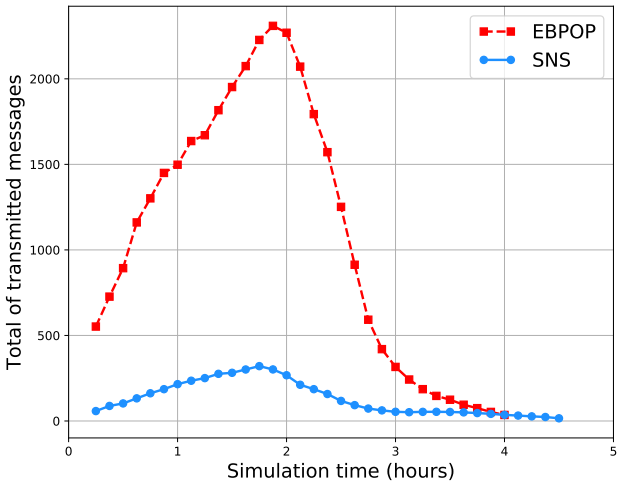


Figure 5.8: Transmitted messages

Having accurate knowledge about traffic conditions is essential to understand the traffic dynamics and enable good traffic management. Figure 5.9 shows the results for the traffic knowledge accuracy in each re-routing interval for both EBPOP and SNS. EBPOP provides an upper bound close to 100% for all re-routing steps, considering that the server knows the current position and velocity of all vehicles. However, as in SNS, the server receives traffic estimations of each sub-region, which is cooperatively built by the vehicles. It does not precisely know the position and velocity of all vehicles. Consequently, this decreases its knowledge about traffic conditions on the roads. In particular, in SNS, as the density increases, the accuracy decreases.

Nevertheless, it still reaches an accuracy higher than 94% for all re-routing steps, which is a suitable accuracy for providing good traffic management. In this scenario, we can see the efficiency of the traffic reporting mechanism employed by SNS, which produces low overhead and provides accurate knowledge about traffic conditions. On the other hand, EBPOP produces an overhead more than 5 times higher than SNS in order to increase

less than 5% of the traffic knowledge (see Figures 5.8 and 5.9).

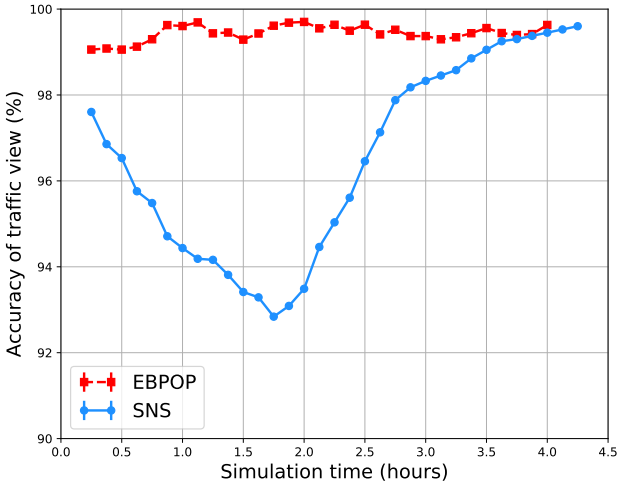


Figure 5.9: Accuracy of the traffic knowledge

To evaluate the time complexity, we considered each solution's CPU time. For this evaluation, we counted the amount of time spent in each re-routing step using an Intel(R) Core(TM) i5-5257U CPU with 2.70 GHz. Figure 5.10 shows the results for this evaluation. As was expected, EBPOP presents the highest CPU time for all re-routing steps, which is a consequence of its centralized architecture since the server needs to compute the alternative routes for all intended vehicles. Compared to EBPOP, SNS reduces by approximately 99% the CPU time (i.e., the complexity time of the re-routing algorithm) due to its hybrid architecture and its efficient cooperative re-routing algorithm. Finally, it is important to notice that, for EBPOP, the CPU time is directly related to traffic density. Therefore, it potentially can introduce high latency to the system in high-density scenarios and degrade its performance. On the other hand, in SNS, its CPU time has no relation to traffic density because it offloads the re-routing algorithm in each vehicle, consequently providing better system scalability.

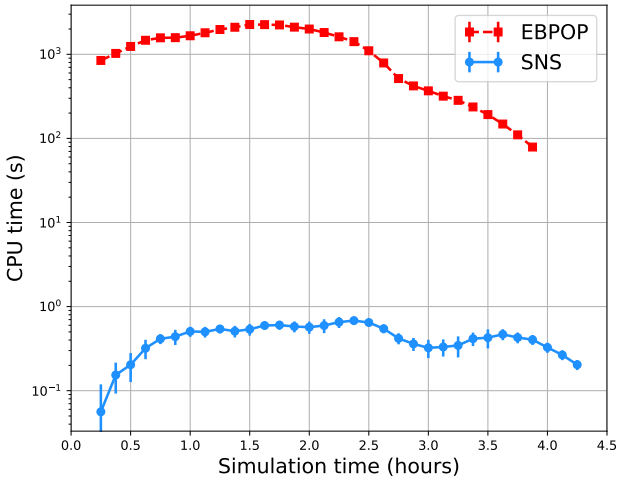
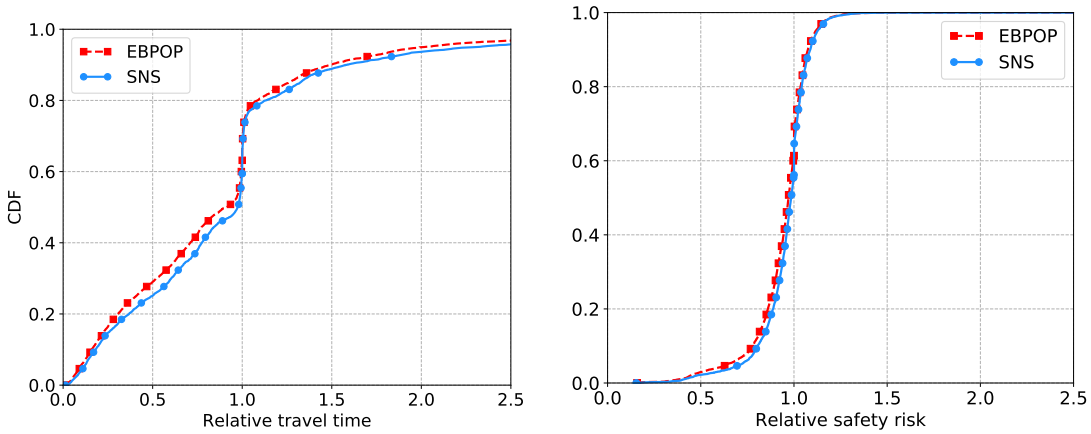


Figure 5.10: Time complexity



(a) Relative travel time

(b) Relative safety

Figure 5.11: Traffic and safety risk results comparing EBPOP and SNS

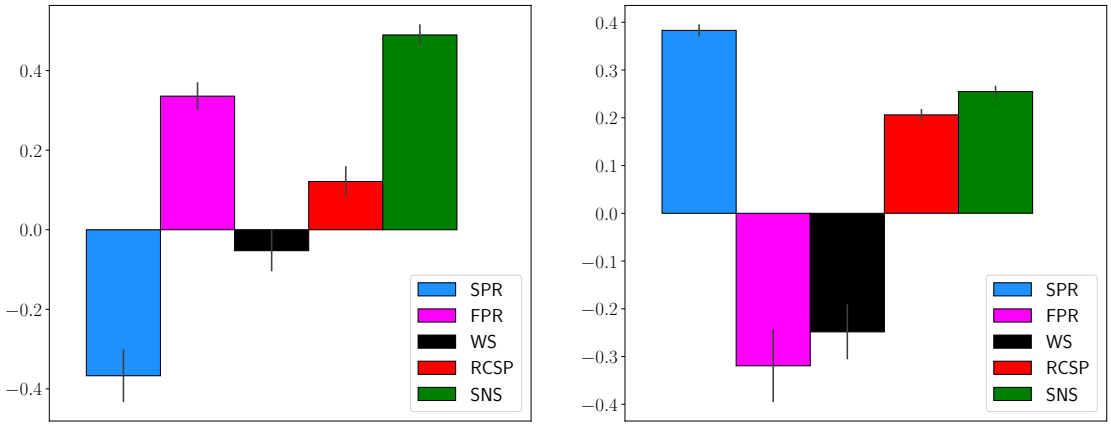
The efficiency of the cooperative re-routing algorithm employed by SNS can be seen by comparing the relative travel time and relative safety risk results. Figure 5.11(a) shows the relative travel time for EBPOP and SNS as a Cumulative Distribution Function (CDF). Compared to the No Re-routing approach, both solutions improve the mobility and safety of the majority of the vehicles. For instance, both solutions reduce the travel time for 60% of the vehicles (see values lower than 1 in Figure 5.11(a)) while increasing the travel time for less than 15% of them (see values greater than 1 in Figure 5.11(a)). Both solutions reduce the safety risk for 80% of the vehicles regarding the relative safety risk. In this context, even with its limited knowledge about the alternative routes taken by the other vehicles (since in SNS vehicles do not know the route taken by all vehicles), SNS can still deal with vehicular mobility properly while improving the safety for drives passengers. This is due to its efficient traffic reporting mechanism, which provides accurate traffic knowledge to improve mobility. It is worth noticing that, for the impaired vehicles, the increase in their travel time is lower than 5 minutes.

We learned three lessons from the results: *(i)* each vehicle only needs to know the route taken by the vehicles in its vicinity to improve the vehicular traffic mobility properly, since the global information about all the vehicle routes brings minimal benefits; *(ii)* SNS is slightly less effective than EBPOP because it misestimates the traffic condition in some road segments, but, the benefits of providing lower overhead and complexity time and higher system scalability overcome this generally acceptable performance loss; and *(iii)* the efficient performance of offloading the re-routing computation in each vehicle paves the way to an efficient, personalized safety-based re-routing, enabling that each vehicle selects the most relevant risks that it wants to avoid.

5.5.3 SNS vs Literature Solutions

In order to evaluate the trade-off between traffic efficiency and safety, we compared the performance of SNS against: *(i)* Safest Path Re-routing (SPR); *(ii)* Fastest Path Re-routing (FPR); *(iii)* Weighted-Sum (WS); and *(iv)* Resource Constrained Shortest

Path (RCSP). It is important to stress that SPR and FPR are both analog to the following literature solutions [33, 34], which focus on re-routing vehicles based on the fastest route and safest one, respectively, while WS and RCSP are both multi-objective optimization approaches.



(a) Average percentage of improvement in relative travel time

(b) Average percentage of improvement in relative safety

Figure 5.12: Results of the trade-off analysis between mobility and safety.

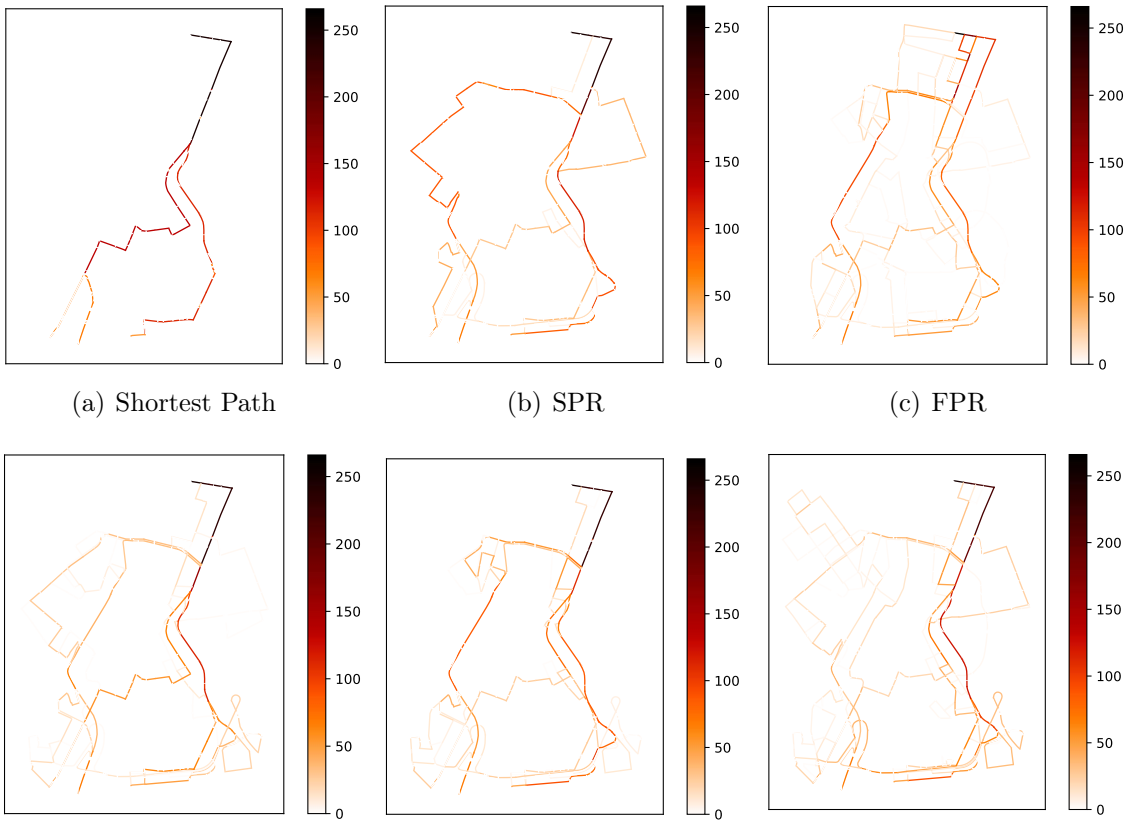
Figure 5.12 shows the results for the relative travel time and relative safety risks as an average percentage of improvement. As expected, since SPR and FPR do not have both pieces of knowledge (e.g., traffic conditions and dangerous areas), they can improve just one metric. In particular, SPR increases the average travel time by about 40%, while FPR reduces the average travel time by about 35% (see Figure 5.12(a)). However, due to the safety knowledge of SPR, it improves average safety by approximately 40%. Consequently, as a result of the lack of that knowledge, FPR increases the average safety risk by approximately 30% (see Figure 5.12(b)). However, due to the lack of both knowledge, those solutions can either recommend dangerous or congested routes. Consequently, the route recommended can be slower or more dangerous than the shortest one.

Thanks to both pieces of knowledge, WS and RSCP can recommend safer routes and still maintain good traffic flow. However, since WS does not employ any constraint for recommending the alternative route, the safety penalty suffered by the vehicles is dramatically higher than the safety penalty in RCSP. As it can be seen, on average, WS increases the risk by 30%, while RCSP decreases by 20% (see Figure 5.12(b)). It is important to notice that RSCP presents a worse travel time result than FPR because a risk constraint gives the limitation of the alternative route. In some cases, the fastest route will not be feasible.

Although the described solutions decrease the travel time for some cases, they are far from a desirable multi-objective re-routing solution due to their deterministic approach. Many vehicles with the exact origin and destination will have the same recommendation. They potentially create bottlenecks in the transportation infrastructure, consequently decreasing the overall traffic efficiency. As it can be observed in Figure 5.12 all those

solutions do not have the most significant improvement in the average travel time.

The efficiency of SNS comes up when comparing the trade-off between mobility and safety. It can decrease the average travel time by 45% while decreasing the average safety risk by approximately 30%. Moreover, as it balances the traffic flow over a set of feasible routes, it decreases the probability of creating different congestion spots. Hence, it reaches a more significant improvement in the average travel time, presenting an increase of 15% higher than the fastest re-routing approach. However, such improvement can be substantial in large scenarios with many vehicles with exact origins and destinations, such as rush hours in large cities. Naturally, the efficiency of the deterministic approaches will also be minimized.



(a) Shortest Path (b) SPR (c) FPR

(d) WS (e) RCSP (f) SNS

Figure 5.13: Set of roads which compose the route of each vehicle according to each approach

To evaluate the traffic flow balancing of each solution, we have selected a sample of 250 routes with the same (s-t) pair, and we analyzed the route recommendation of each solution to every single route of this set.

Figure 5.13 shows routes recommended for all vehicles using each solution, the road taken for each vehicle is colored on the map according to its frequency. As expected, the Shortest Path approach shows the lowest number of different routes since it does not consider the traffic conditions.

On the other hand, as the other solutions consider the traffic conditions, they increase alternative paths since the traffic conditions are dynamic. Therefore, route recommenda-

tions can change according to the traffic conditions, but they are still deterministic. Thus, as presented in Figures 5.13(c), 5.13(d) and 5.13(e) they potentially create bottlenecks in different roads (see the dark red roads).

Nevertheless, SNS's traffic flow balancing can be seen when comparing the number of roads and their frequency against the other solutions. Hence, since it considers the risk for balancing the traffic, which considers both the criminal activity and the traffic condition, SNS is less likely to create different congestion spots than the other solutions while dealing with mobility and safety issues.

5.5.4 Personalized Safety Risk with Spatiotemporal Analysis

Different drivers and passengers (i.e., vehicles) potentially have different preferences according to each crime's safety risk. These preferences can vary depending on each driver's profile, gender, age, also depending on the day and time [**?**]. Hence, based on each preference of each vehicle, the safety risk potentially changes for them, even for vehicles with the exact origin and destination but with different preferences according to the type of crime they want to avoid.

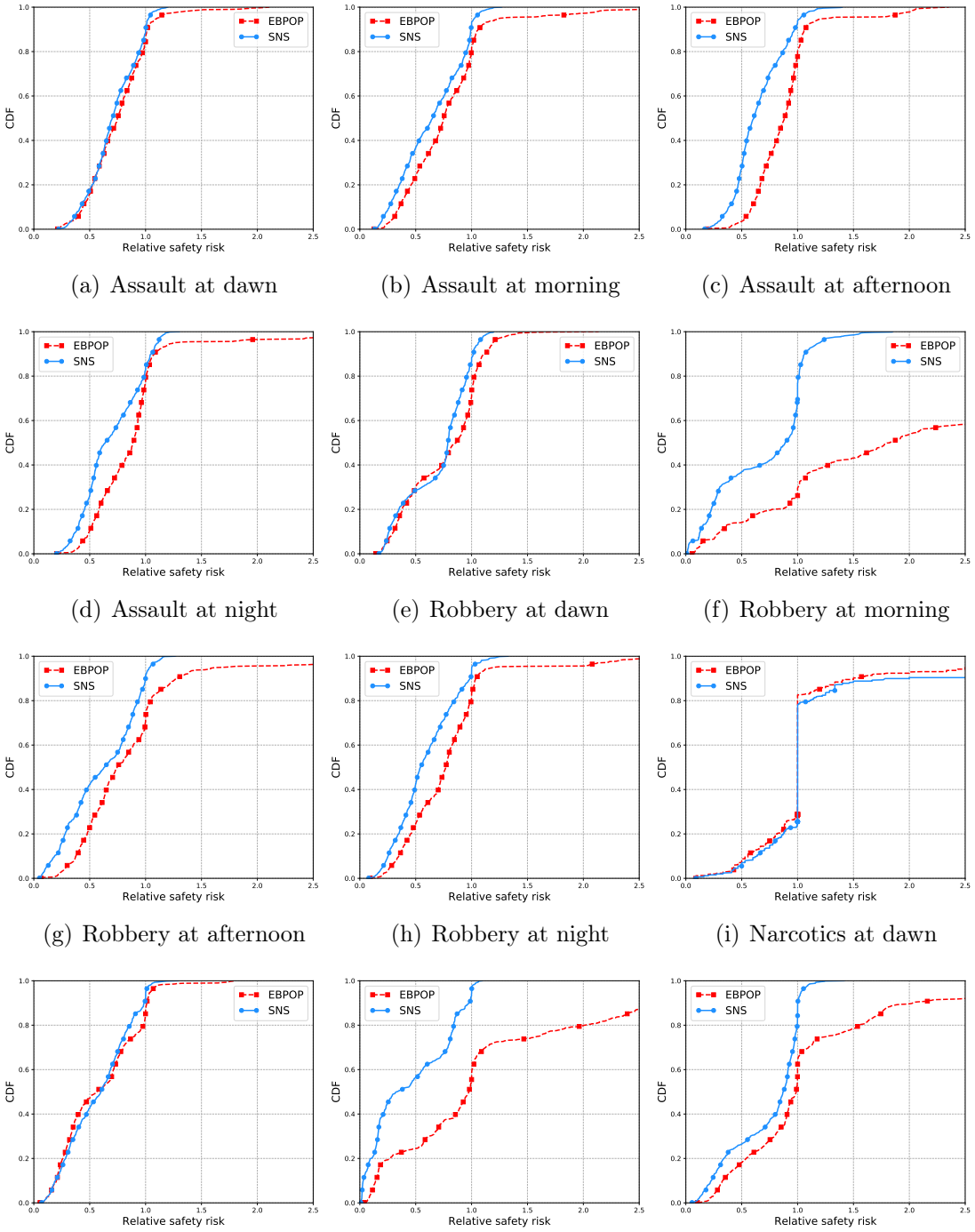
Since SNS enables personalized re-routing and also explores the spatial and temporal information about dangerous areas, with this analysis, we want to assess how these preferences and the spatiotemporal correlation of criminal activities can impact the safety of the drives and passengers. To do so, we differentiate the days into *Business days* and *Weekend* and also segmented each day into four periods named as: *(i)* Dawn, from 00:00 to 05:59; *(ii)* Morning, from 06:00 to 11:59; *(iii)* Afternoon, from 12:00 to 17:59; and *(iv)* Night, from 18:00 to 23:59. In this way, by randomly selecting a day and time, we predicted the criminal density for each type of crime using the RNN and performed the vehicular traffic re-routing considering each crime preference.

Table 5.3: Average route safety risk for assault-related crimes on business days and weekend considering different periods of the day.

	Business days				**Weekend**			
	Dawn	Morning	Afternoon	Night	Dawn	Morning	Afternoon	Night
No Re-routing	16.07	14.02	17.03	16.65	11.62	3.55	16.54	18.61
EBPOP	11.04	9.55	13.67	12.87	10.15	3.00	15.07	17.98
SNS	10.76	8.54	10.72	11.00	8.87	2.25	10.52	15.07

Table 5.4: Average route safety risk for robbery-related crimes on business days and weekend considering different periods of the day.

	Business days				**Weekend**			
	Dawn	Morning	Afternoon	Night	Dawn	Morning	Afternoon	Night
No Re-routing	14.05	0.65	13.10	14.58	11.78	11.44	10.88	10.51
EBPOP	9.88	0.63	9.91	9.43	8.22	6.76	7.91	8.46
SNS	9.47	0.10	8.24	8.20	6.19	6.28	5.66	6.88



(a) Assault at dawn (b) Assault at morning (c) Assault at afternoon

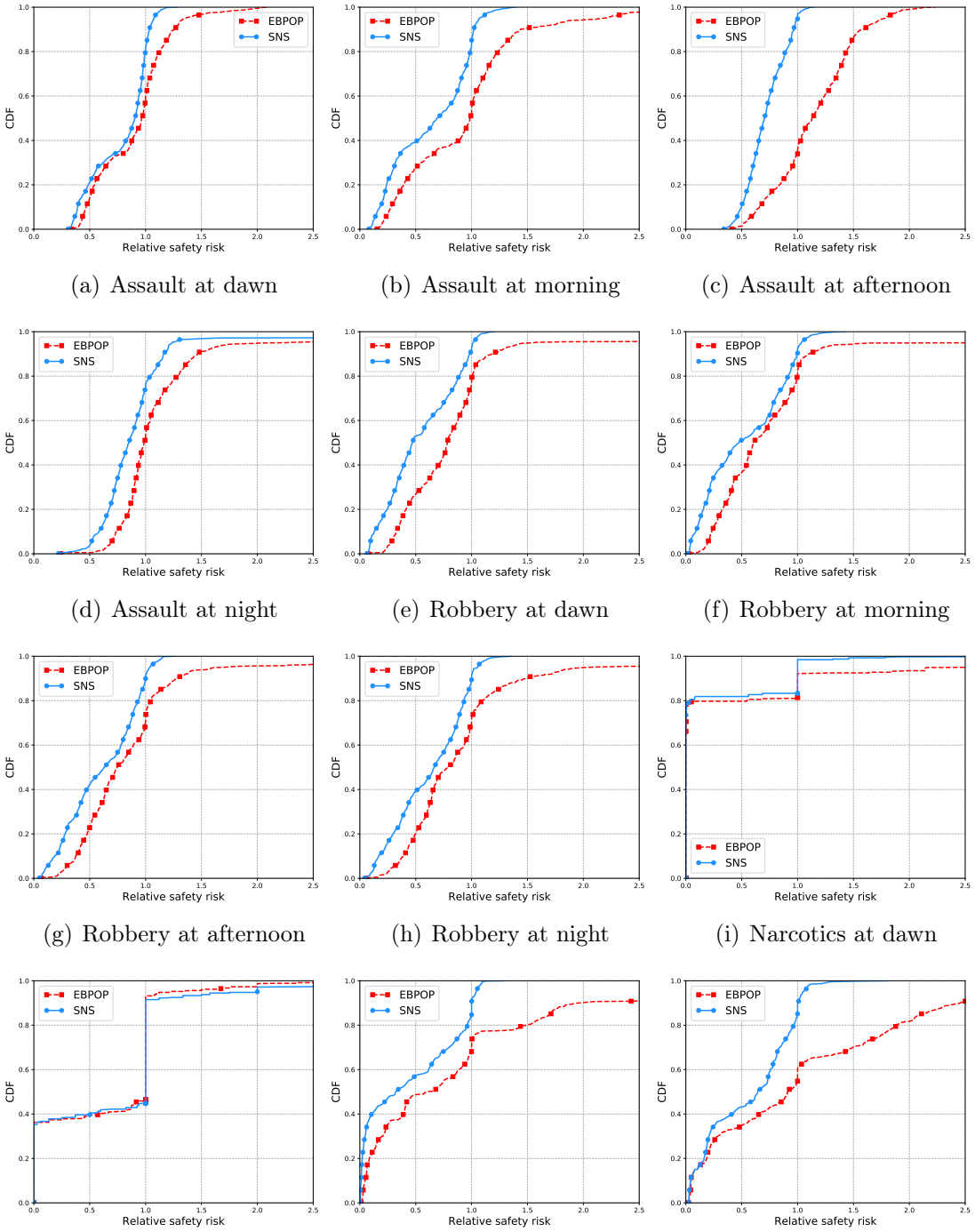
(d) Assault at night (e) Robbery at dawn (f) Robbery at morning

(g) Robbery at afternoon (h) Robbery at night (i) Narcotics at dawn

(j) Narcotics at morning (k) Narcotics at afternoon (l) Narcotics at night

Figure 5.14: Comparison of relative safety risk results for EBPOP and SNS based on different periods of the day and type of crimes during business days

Figure 5.14 shows the relative safety risk results for each period of the day for the type of crime during business days comparing SNS and EBPOP, while Figure 5.15 shows the same results during weekends. Also, Tables 5.3, 5.4 and 5.5 show the average route safety risk result (e.g., r_P) for each solution considering each type of crime. As it can be seen, SNS outperform EBPOP for all days and periods considering the different safety risks provided by each criminal activity (see Figure 5.14 and Figure 5.15). This results from



(a) Assault at dawn (b) Assault at morning (c) Assault at afternoon

(d) Assault at night (e) Robbery at dawn (f) Robbery at morning

(g) Robbery at afternoon (h) Robbery at night (i) Narcotics at dawn

(j) Narcotics at morning (k) Narcotics at afternoon (l) Narcotics at night

Figure 5.15: Comparison of relative safety risk results for EBPOP and SNS based on different periods of the day and type of crimes during weekends

the preference-based re-routing employed by SNS. Vehicles just consider the relevant risk for them, while in EBPOP, the system just knows the overall risk, not considering which criminal activity affects each part of the city according to the day and time. Thus, if another criminal activity has a higher density in some area than the other criminal activities, vehicles that have a preference about safety risks with lower density will be impaired because EBPOP can guide them towards regions that they do not want to avoid other

Table 5.5: Average route safety risk for narcotics-related crimes on business days and weekend considering different periods of the day.

	Business days				**Weekend**			
	Dawn	Morning	Afternoon	Night	Dawn	Morning	Afternoon	Night
No Re-routing	0.26	10.35	3.26	4.86	0.14	0.07	4.49	3.61
EBPOP	0.25	4.78	2.81	4.58	0.04	0.06	3.35	2.97
SNS	0.24	4.54	1.31	3.19	0.01	0.05	1.73	1.72

areas with higher density, consequently increasing the safety risk when considering their preferences. However, SNS vehicles just avoid the regions and the safety risks relevant to them, which improves their safety according to their preference.

In particular, SNS reduces the average safety risk by approximately 30% compared to EBPOP, considering the safety risk provided by different criminal activities and their spatiotemporal correlation. This results from the accurate criminal density prediction provided by the RNN and the efficiency of the cooperative re-routing algorithm offloaded in each vehicle, which enables the personalized re-routing to improve the safety of drivers and passengers based on their preferences.

5.6 Chapter Conclusions

This chapter presented SNS, a non-deterministic multi-objective re-routing system for improving traffic efficiency while improving the safety of drivers and passengers. To do so, SNS extracts knowledge about traffic conditions and safety risks over the city. On the one hand, the knowledge about traffic conditions is obtained based on the vehicles' traffic information. On the other hand, to extract the knowledge about safety risks over the city, SNS proposes an algorithm to discover the safety risks based on criminal activities. The key idea of the algorithm is to identify hotspots for criminal activities using a KDE for each type of crime. In this way, by segmenting the set of crimes that happened throughout the day considering a time window of one hour, SNS explores the spatiotemporal correlation. This means that the system can identify the regions that are more likely to provide higher chances for criminal activities and when those regions will become dangerous.

With both pieces of knowledge, SNS employs non-deterministic multi-objective re-routing to improve traffic efficiency and the safety of drivers and passengers. The re-routing algorithm reduces the problem of creating additional congestion spots by distributing the traffic flow over the routes in the Pareto-set (i.e., routes that improve both metrics, mobility, and safety). Besides, SNS explores the spatiotemporal correlation of criminal activities and enables personalized re-routing (in which the drivers can choose which type of crimes they want to avoid) to push the system's efficiency further.

Nevertheless, SNS does not care about future changes in urban dynamics, which means that future changes in either the traffic efficiency or the safety dynamics might degrade the efficiency and the reliability of the routes previously computed. SNS relies on periodic re-routing to minimize such a problem. However, this procedure increases network usage and computational efforts of the system because the system needs to gather the traffic

information over the scenario and compute new routes to each vehicle during each re-routing phase. Knowing when the changes in the urban dynamics will happen beforehand is essential to recommend more reliable and efficient routes. In this way, the following chapter will present methods and algorithms to predict future urban dynamics and use those predictions during route planning to provide more efficient and reliable routes.

Chapter 6

The Sequential Decision-making Under Uncertainty in Vehicular Route Planning

Research question 3: *How to consider the future changes in urban dynamics during the re-routing to plan a more efficient and reliable route?*

Previous chapters have shown that route recommendation systems are key solutions for dealing with vehicle routing problems, from simple dynamic routing to improve the overall traffic efficiency to more complex solutions that consider multiple objectives and allow different customization based on users' preferences. However, future changes in urban dynamics might degrade the efficiency of routes previously planned by these systems. Therefore, to tackle this problem, literature solutions rely on a periodical re-routing process [54, 22, 29, 33], which means that if a route previously computed becomes inefficient, the system computes a new route based on an updated observation of the scenario, consequently increasing the computation efforts and also network costs [67, 18]. In this scenario, considering future urban dynamic changes during the route planning can push the efficiency of route recommendation systems even further.

Considering future changes in urban dynamics during route planning can be considered sequential decision-making under uncertainty. Also, the system needs to know or predict future changes in the environment to enable reliable route planning. In this way, this chapter introduces VTq (Vehicular Traffic management with Q-learning), an efficient routing system for dealing with sequential decision-making under uncertainty problems. The key idea of the system is to predict future urban dynamics using a Long-Short Term Memory (LSTM), which is an efficient recurrent neural network for time series forecasting. Then, implementing a reinforcement learning-based routing algorithm to learn the most efficient and reliable route considering the future dynamics. In summary, the algorithm learns how to build the most efficient path throughout a trial-and-error approach by interaction with the environment based on the predictions provided by the LSTM. This combination enables the system to take *pro-active decisions* in such a *reactive environment*.

By the end of the chapter, we will have provided solutions to deal with the following issues: *(i)* how to provide accurate predictions about future urban dynamics considering historical data; and *(ii)* how to use those predictions during the route planning to compute

more efficient and reliable routes.

The rest of this chapter is organized as follows. Section 6.1 provides the system overview describing architecture and mechanisms used. Section 6.2 describes the prediction mechanism used by the system, while Section 6.3 describes the route planning algorithm that considers future changes in the urban dynamics. At last, Section 6.4 presents the performance analysis of the proposed system and Section 6.5 concludes the chapter.

6.1 System Overview

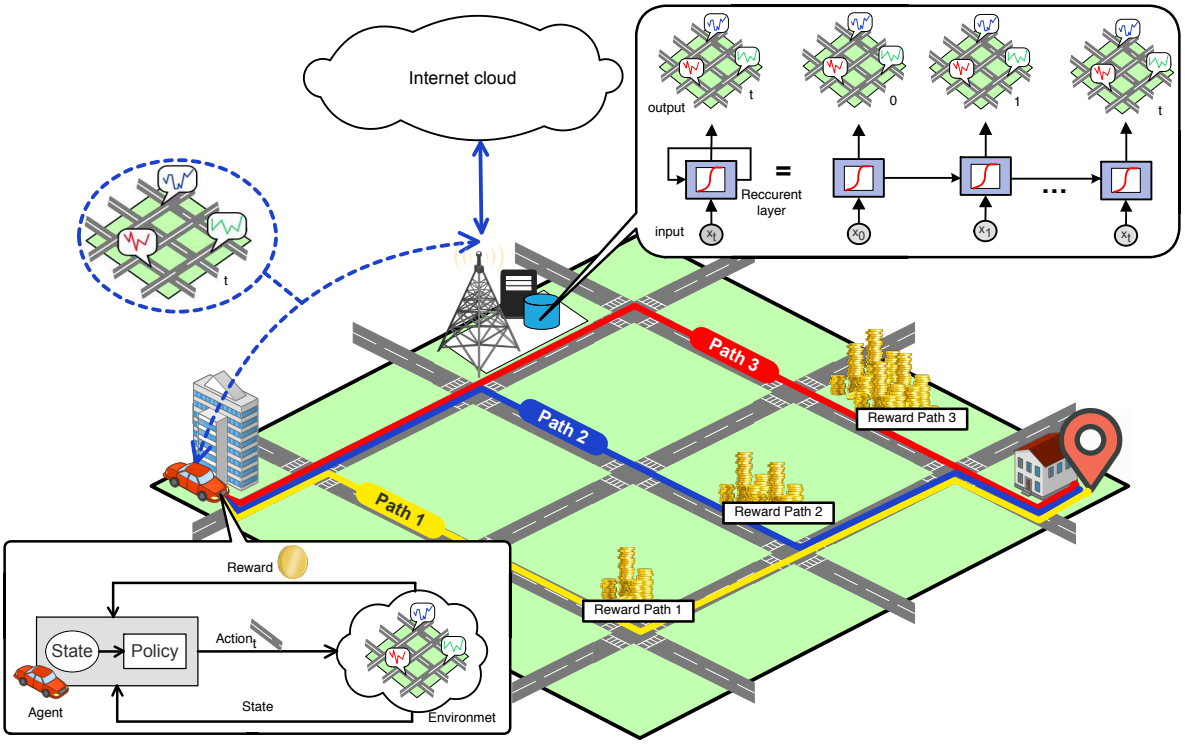


Figure 6.1: Reliable path planning for TMS considering future urban dynamics

Figure 6.1 presents an overview of the architecture and operation of the proposed system. First, when a vehicle is to start its journey, it requests predictions of the desired urban dynamics such as traffic conditions or safety risks to an edge server responsible for the region where the vehicle is. In this way, the last prediction data is returned to the vehicle using the LSTM located on the edge server.

Thus, when receiving this information, the vehicle updates the graph's values that represent the scenario, and the reinforcement learning algorithm plans the route. It is important to note that sharing this information between the vehicle and the edge server does not introduce latency or reduce system efficiency due to the protocols for sharing and requesting information implemented by VTq (more details on this process can be found at [17, 24, 15]).

6.2 Predicting Future Urban Dynamics

Current path planning approaches do not guarantee the reliability of the planned path concerning future changes in the urban environment. Considering the massive amount of data provided by the urban environment and its inter-correlation, LSTM can play an essential role by significantly improving the prediction of urban dynamics by exploring long sequences of data and their inter-correlation to sharp its predictions, enabling more reliable decision-making and urban planning services.

The key idea in predicting future urban aspects is to learn the dynamics in each neighborhood and/or road segment to know in advance when and where future changes in the dynamics will happen, considering its spatiotemporal correlation. The system needs to extract knowledge from previous events to learn such dynamics, [57]. In this context, several machine learning techniques have been used to pull out the knowledge obtained from past events such as linear regression, Markov models, support vector machines, and neural networks [67]. However, these approaches potentially have issues considering spatial and temporal setup since they can not distinguish the spatial and temporal correlations accurately [47, 57].

One important task for handling with LSTMs is the input data, which needs to be pre-processed to work as temporal sequences. In most cases, the data gathered from urban dynamics such as traffic conditions and safety risks are not proper for working with LSTM. Thus, it needs to be reshaped to fit the requirements. For instance, let $X = \{x_1, x_2, x_3, \ldots, x_n\}$ be a sequence presenting some urban dynamics. Let's assume that we want to build an LSTM to predict the dynamics one step further $t = 1$. In this case, to train the LSTM, we need a ground truth to fit the model. Thus, let $Y = \{y_1, y_2, y_3, \ldots, y_n\}$ represent the ground truth from which the LSTM will learn from. Here, Y is obtained from shifting X by t elements, which means that Y is X one step further. Moreover, to reshape X into sequences, we need to define the desired length of each sequence. Let's define a *sequence length* $= 3$; in this way, X will be reshaped into slices of 3 elements. For the sake of illustration, Figure 6.2 shows the reshaping process for a input that $X = \{0, 1, 3, 4, 5, 6, 7, 8, 9\}$ with prediction window $t = 1$ step and *sequence length* $= 3$.

Another essential feature of LSTM is the *batch size*, which refers to the number of training samples utilized per iteration or epoch. An *epoch* defines the total number of iterations to train the whole dataset based on the size of the batch. The input data is divided into many batches depending on the batch size and is fed into the LSTM. Once it has computed the result of a single batch, the network calculates its gradient and updates its weights. The batch size is one of the most important hyperparameters to tune in modern deep learning systems. A larger batch size allows computational speedups from the parallelism of GPUs to train the model. However, depending on how large is the batch size it will lead to poor generalization [43, 47]. It is worth noticing that using a batch equal to the entire dataset guarantees convergence to the objective function's global optimum. However, this is at the cost of slower, empirical convergence to that optimum. On the other hand, using smaller batch sizes empirically showed faster convergence to reasonable solutions. This is intuitively explained by the fact that smaller batch sizes allow the model to start learning before seeing all the data. The downside of using a

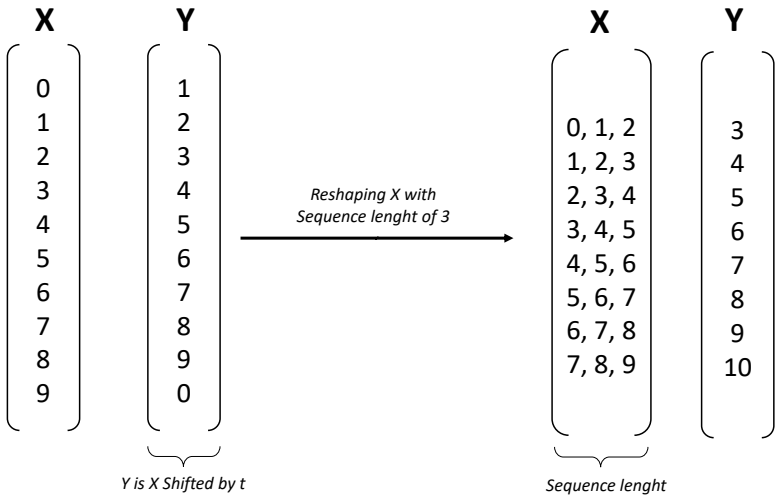


Figure 6.2: Reshaping input data for LSTMs.

smaller batch size is that the model is not guaranteed to converge to the global optimal. Figure 6.3 shows an example of a training iteration with a batch size of 16 and sequence length of 24 representing what the LSTM sees in each iteration.

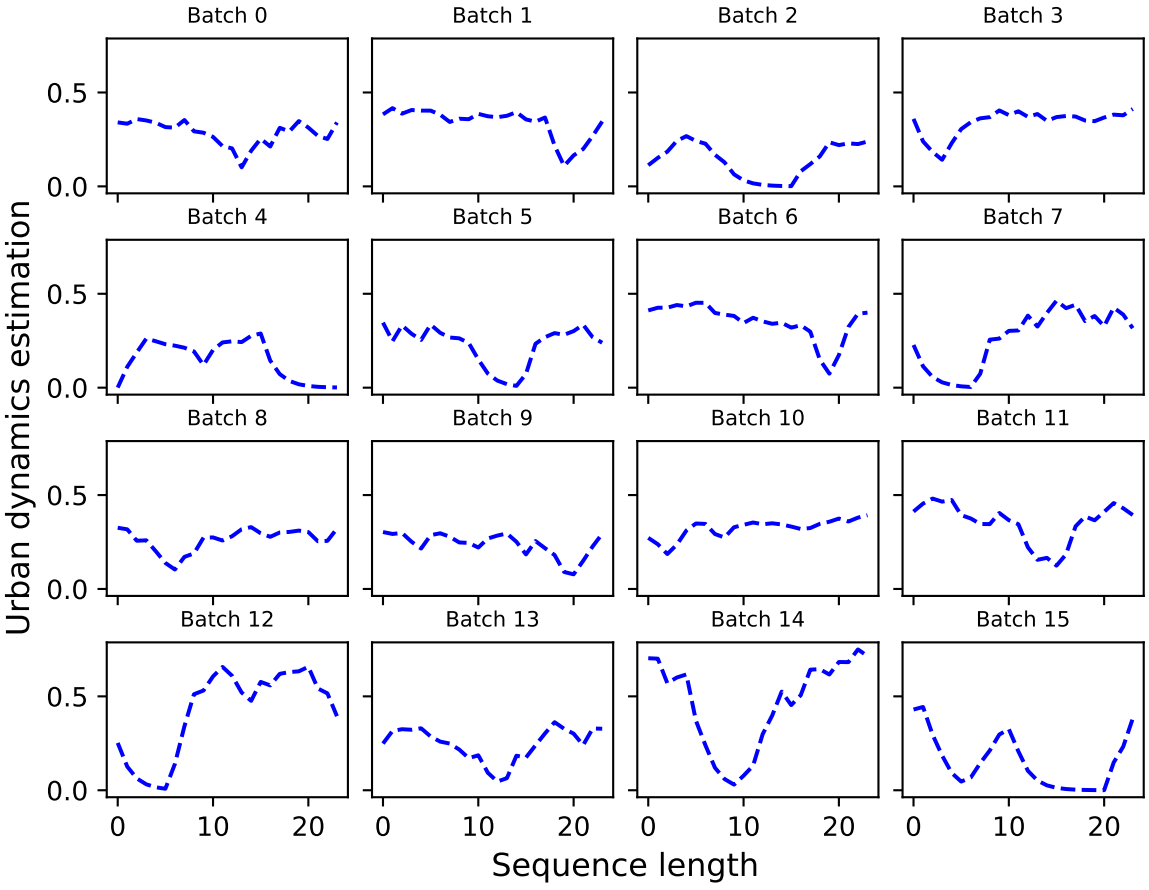


Figure 6.3: Batch representation

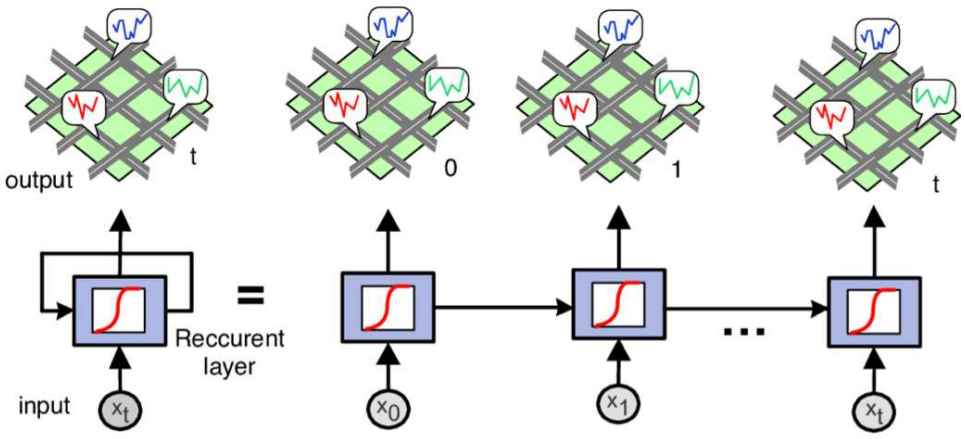


Figure 6.4: LSTM employed by VTq.

Figure 6.4 shows the LSTM employed VTq. The LSTM receives an input x^t representing a set of tuples (e.g., timestamp, uv, r_{uv}, urban dynamic estimation). For each road segment and estimation, it employs an LSTM as a recurrent layer with a tanh activation function. We illustrate how the recurrent layer works by unrolling the past predictions $x_0, x_1, \ldots, x_t$, which are used as long-term memory to explore each prediction's internal correlation. Finally, the output is represented as the future dynamic of each road considering the desired urban dynamic. It is worth noticing that several hyper-parameters might impact the performance of the LSTM including, the sequence length, batch size, number of layers, and epochs.

6.3 Reliable Path Planning based on Future Dynamics

The main reason for the unreliability in current path planning algorithms is that vehicles are not aware of when or where some congestion will occur and how long it will last. Thus, current approaches can guide vehicles to roads that will become congested shortly or also cause congestion in some roads due to the implementation of deterministic path planning approaches [54]. With the assistance of LSTM for providing a better understanding of future urban dynamics, new approaches can be built to provide a more reliable path planning. To do so, we modeled the route planning problem as a finite Markov Decision Process (MDP).

6.3.1 Route Planning as a Finite MDP

Consider the environment as a road network represented as a finite MDP. The set of states S, is represented by the intersections in the scenario, and the set of actions A is represented by the road segments connecting two intersections. Let P represents the transition probability to each state obtained from the road network. Each action $a \in \mathcal{A}$ taken at time t from a state $s \in \mathcal{S}$ gives an immediate reward r which is represented by the urban dynamics and leads to a new state $s' \in \mathcal{S}$ which is denoted by $p(s', r \mid s, a)$. Given the current state s, a policy π is a probability distribution over possible actions

at state s, denoted as $\pi(a \mid s)$. Then, based on a policy, the agent can navigate in the environment (i.e., go from one state to another) and get rewards through each transition. In other words, a policy π defines a route in the environment in which a vehicle can travel through. The value of each state in a policy π is defined as:

$$v_\pi(s) = E_\pi \left(\sum_{t=1}^{\infty} \gamma^{t-1} R_t \mid S_1 = s \right) \tag{6.1}$$

and the value of each pair of state and actions as:

$$q_\pi(s, a) = E_\pi \left(\sum_{t=1}^{\infty} \gamma^{t-1} R_t \mid S_1 = s, A_1 = a \right) \tag{6.2}$$

In this context, the problem of finding the most reliable path for each vehicle considering future changes in the urban environment is modeled as:

$$\pi^* = \arg\max_{\pi} v_\pi(s), \tag{6.3}$$

where π^* represents the optimal policy (e.g., path) in the environment considering the rewards received from the sequential pair of state and actions at time t. It is important to notice that the rewards were modeled as negative values to avoid loops during the route planning. Also, episodes are defined to find each policy. The process of finding a policy represents each episode (e.g., a path) from an origin (e.g., state s_0) to a destination (e.g., state s_t).

6.3.2 Temporal Difference Learning for Route Planning

Temporal difference (TD) learning algorithms can learn how to estimate values based on other estimations. Each step in the environment generates a learning example that can be used to bring some value according to the immediate reward and the estimated value of the following state or state-action pair.

TD methods learn their value estimates based on estimates of other values called bootstrapping. They have an advantage over dynamic programming-based algorithms in that they do not require a model of the MDP. Another advantage is that they are naturally implemented in an online, incremental fashion such that they can be easily used in various circumstances. No full sweeps through the entire state space are needed; only along experienced paths values get updated, and updates are effected after each step.

One of the most basic and popular methods to estimate Q-value functions in a model-free fashion is the Q-learning algorithm [43]. The basic idea in Q-learning is to incrementally estimate Q-values for actions based on feedback (i.e., rewards) and the agent's Q-value function. The update rule is a variation on the theme of TD learning, using Q-values and a built-in max-operator over the Q-values of the next state in order to update Q_t into Q_{t+1}:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t) \right) \tag{6.4}$$

The agent makes a step in the environment from state s_t to s_{t+1} using action a_t while receiving reward r_t. The update takes place on the Q-value of action at in the state s_t from which this action was executed.

Q-learning is exploration-insensitive. It means that it will converge to the optimal policy regardless of the exploration policy being followed, under the assumption that each state-action pair is visited an infinite number of times and the learning parameter α is decreased appropriately.

Algorithm 5: Reinforcement learning based path planning with predicted urban dynamics

Input : α // *The Q-learning rate*
$G = (V, E)$ // *Road network graph structure*
X_t // *Set of the predicted urban dynamics*
origin // *origin of the path*
destination // *destination of the path*

Output: Q-table to build the most reliable path

```
1  R_{0,...,max(t)} ← buildRewardMatrix(G, X, Δ_t);
2  Q ← buildQTable();
3  while Q is not converged do
4  |   S_i ← origin;
5  |   t ← timeToStartPath();
6  |   while S_i ≠ destination do
7  |   |   A_t ← getAvaliableActions(S_i, R_t);
8  |   |   a_t ← selectAction(A_t);
9  |   |   Q.update(S_i, a_t, γ, R_t);
10 |   |   S_i ← S_i + a_t;
11 |   |   t ← t + timeOf(a_t, S_i);
12 |   end
13 end
```

Algorithm 5 shows the reinforcement learning procedure executed by the vehicle (i.e., agent) to determine which road will be selected to compose its path (i.e., action) based on the current state of the urban environment, its learning policy (i.e., fastest, safest path, and others), and the expected reward according to the predicted urban dynamics.

To plan the path towards its destination, the vehicle checks the available actions $\mathcal{A}$ at time t based on the current state S_i and the reward matrix R_t. Hereafter, the vehicle selects an action $a_t \in \mathcal{A}_t$ at time t and then updates the Q-table according to the current state S_i, the selected action a_t and the expected reward R_t at time t. The Q-learning updating function can be expressed as the following iterative formulation to maximize the expected reward according to Equation 6.4.

One crucial issue that needs to be handled in route planning systems is computing the new alternative route to the vehicle (e.g., CPU time). Because the computation takes too long, the vehicle may not improve the traffic efficiency since it might receive the new path too late. In this way, two approaches were implemented in the reinforcement learning-based algorithm to improve its CPU time: *(i)* ϵ-greedy action selection; and *(ii)* experience replay.

The **ϵ-greedy** acts to improve the exploration and exploitation trade-off of reinforcement learning algorithm, in which the *exploration* phase relies on the search for new

policies in the environment without caring about their optimally. On the other hand, the *exploitation* phase relies on using the knowledge obtained from the exploration to find the most efficient policy. Thus, the ϵ-greedy algorithm is defined as:

$$\begin{cases} \epsilon & random(A_t) \\ 1-\epsilon & \max(Q(s_t, a_t), \end{cases} \tag{6.5}$$

where it selects a random action in the action space from a state s with probability ϵ to explore. Otherwise, it selects the best available action with probability $1-\epsilon$ to explore. This approach allows the algorithm to not get stuck in local optimum policies.

In addition, experience replay is a technique used to improve the convergence time, in which it uses the knowledge obtained from past/different environments to find the optimal policy in the new environment. In other words, the experience replay acts as transfer learning between different environments.

6.4 Performance Analysis

This section analyzes the performance of the proposed solution regarding the urban dynamics prediction and the route planning considering future dynamics. For this evaluation, we have considered two urban dynamics that can impact route planning decision, which is: *(i)* traffic condition, represented by the travel time on the roads; and *(ii)* safety risks, represented by the criminal density estimated on the roads. It is worth noticing that the traffic efficiency and safety were used as a use case for evaluating VTq. However, other urban dynamics can be naturally used by the system.

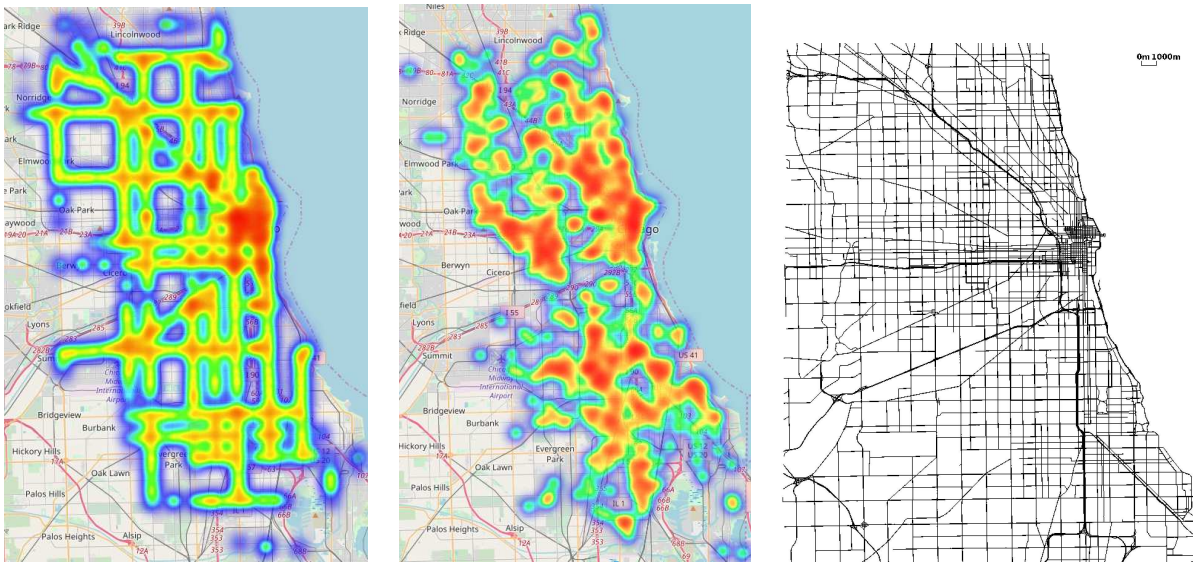
In this way, Section 6.4.1 describes the datasets used in this evaluation. Section 6.4.2 introduces the evaluation methodology. Section 6.4.3 describes the metrics used to evaluate the mechanisms implemented by VTq. Section 6.4.4 analyzes the impact of the batch size and the sequence length during the LSTM predictions. Section 6.4.5 presents the analysis of the predictions of the LSTM for traffic efficiency and safety risks in comparison with other approaches. Section 6.4.6 evaluates the exploration vs. exploitation trade-off and the experience replay methods for improving the route planning algorithm proposed by VTq. At last, Section 6.4.7 compares the performance of VTq in respect to the literature solutions for path planning considering traffic efficiency and safety risks.

6.4.1 Dataset Description

To assess the efficiency of predictions and planning, we use real data on mobility and public safety from Chicago, USA. These data were obtained through the Chicago open data portal[1]. The data can be found between February 2018 until December 2019. Figure 6.5 shows an example of the data used for October 20, 2019, for both mobility and safety.

Traffic dataset is composed of about 1000 roads with estimations of traffic congestion (i.e., the current speed of each road) every 10 minutes. Each estimation is based on real-

[1] `https://data.cityofchicago.org/`



(a) Traffic dataset (b) Safety risk dataset (c) Chicago road network graph

Figure 6.5: Example of the datasets and the scenario used.

time information provided by GPS probes from the Chicago Transit Authority (CTA) buses.

Safety dataset is composed of the set of crimes that happened in the evaluated period. To analyze the same areas as the traffic dataset, we used the starting and ending points (i.e., latitude and longitude) of each available road in the traffic dataset and computed the crime density of each road to estimate its safety risk by using the Gaussian Kernel Estimation method with the same granularity of the traffic dataset. Let C be the set of crimes that happened in the city in a specific time window. The criminal density of each road is computed as:

$$\theta(p) = \frac{1}{|C|} \cdot \sum_{c \in C} \frac{1}{h\sqrt{2\pi}} e^{\left(-\frac{1}{2}\left(\frac{||c-p||}{h}\right)^2\right)}, \tag{6.6}$$

where $||c-p||$ is the Euclidean distance between points c and p, and h is the used bandwidth. The bandwidth h defines the Gaussian kernel spread, and controls the smoothness of the estimated density, which was defined based on [59]. Once the density function is estimated, the safety risk of each road uv is computed based on average crime density of $\theta(u) + \theta(v)$.

6.4.2 Evaluation Methodology

This use case analyzes the improvement of urban dynamics prediction and path planning services by using a recurrent neural network and reinforcement learning, respectively. The neural network was implemented using TensorFlow v1.12, while the reinforcement learning was implemented using SciKit Learn v0.20.2.

For the analysis of urban dynamics prediction, we used a train-test split with 90% of the datasets for training and 10% for testing computed using 10-fold cross-validation. The

number of cells of the LSTM was defined based on a set of experiments which showed that the size of 256 cells in its recurrent layer provided the best performance in *Bidirectional* manner. It was trained for 100 epochs to speed up the training. We used the Early Stopping method, which monitors the training progress within epochs by checking the computed accuracy of each training epoch. If the fluctuation of accuracy over the last 10 epochs is less than the $\Delta_{min} = 0.1$, we stop the training. The batch size and the sequence length were defined based on an exploratory analysis described in Section 6.4.4. We set the Q-learning rate (α) and discount factor (γ) to 0.01 and 1, respectively, to prioritize rewards in the distant future. Each prediction represents the traffic and safety dynamics of the urban environment for the next hour with a granularity of 10 minutes considering an input sample of the previous weeks and hours' dynamics. For the traffic analysis, the predictions represent the average speed dynamics of each road. In contrast, for the safety analysis, the prediction means the crime density (i.e., estimated safety risk) of each road.

6.4.3 Metrics

This section describes the metrics used to evaluate the LSTM predictions and also to evaluate the Q-learning-based routing algorithm. To evaluate the LSTM predictions, the following metrics were assessed:

- **Mean Squared Error (MSE)** measures the average of the squares of the errors, that is, the average squared difference between the estimated values (y_i') and the actual value(y_i). MSE is a risk function, corresponding to the expected value of the squared error loss. MSE is a measure of the quality of a predictor, it is always non-negative, and values closer to zero are better.

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(y_i - y_i')^2$$

- **Mean Absolute Error (MAE)** measures errors between paired observations expressing the same phenomenon. Examples of y_i versus y_i' include comparisons of predicted versus observed. The mean absolute error uses the same scale as the data being measured. This is known as a scale-dependent accuracy measure and, therefore, cannot be used to make comparisons between series using different scales. The mean absolute error is a common measure of forecast error in time series analysis

$$MAE = \frac{1}{n}\sum_{i=0}^{n} \mid y_i - y_i' \mid$$

- **Correlation ($\mathbf{R^2}$ score)** measures the fit of a generalized linear statistical model, such as simple or multiple linear regression, to the observed values of a random variable. It expresses the amount of variance in the data that is explained by the linear model. Thus, the higher the score is, the more explanatory the linear model is, i.e., the better it fits the sample

$$R^2 = 1 - \frac{\sum_{i=0}^{n}(y_i - y_i')^2}{\sum_{i=0}^{n}(y_i - mean(y))^2}$$

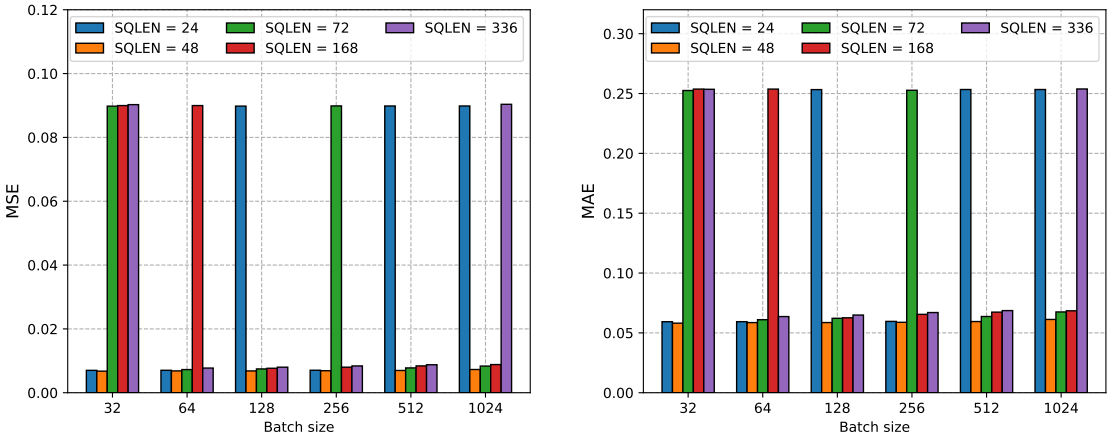
On the other hand, to evaluate the efficiency of the route planning algorithm the following metrics were used:

- **Travel time** is the total time spent by each vehicle to travel its entire journey considering the route planned according to the re-routing algorithm. This metric summarizes the traffic efficiency of the routing algorithm in which the lower the travel time is, the better is the traffic efficiency;
- **Safety risk** is accumulated by each vehicle to travel its entire journey considering the route planned according to the re-routing algorithm. This metric summarizes the safety efficiency of the routing algorithm in which the lower the safety risk, the better is the overall safety of the system;
- **Accumulated reward per episode** an episode is the process of finding the path from the initial state to the ending state in the graph (i.e., the procedure of planning a path for a vehicle). The accumulated reward is expressed negatively due to modeling constraints (to avoid cycles during the path planning). It is used to show the efficiency of the Q-learning algorithm the higher the accumulated reward, the better the performance of the algorithm (until its convergence). The goal is to achieve the best using as few episodes as possible;
- **CPU time** is the time spent by the re-routing algorithm to compute the new alternative route to the intended vehicles. A longer CPU time potentially introduces latency to the system, consequently degrading its overall performance.

6.4.4 Evaluation of the Batch size and Sequence Length of the LSTM

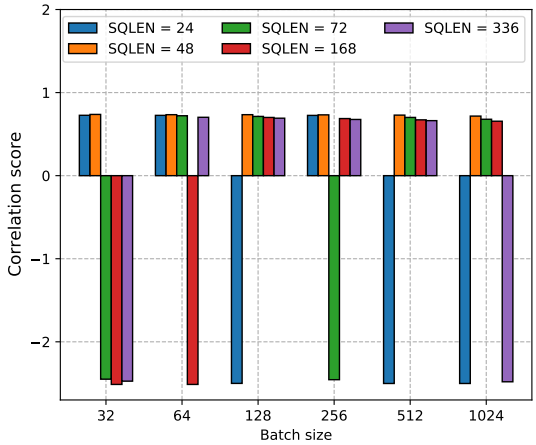
Figure 6.6 shows the results for the batch size and sequence length analysis for the traffic condition dataset. For this analysis, the following values of the batch were used: batch size = $\{32, 64, 128, 256, 512, 1024\}$, and the values of SQLEN = $\{24, 48, 72, 168, 336\}$ for the sequence length representing the time series from one day to two weeks. Besides, to analyze the performance of LSTM, the following metrics were used: MSE, MAE, and correlation score.

As it can be seen, different configurations of batch size and sequence length lead to different performances, but the configuration with a batch size of 64 and sequence length of 48 reached the best results for all assessed metrics. This means that this configuration provides the best representation of the dataset, allowing the LSTM to sharp its predictions. On the other hand, larger batch sizes jointly with longer sequence lengths cannot provide a good representation of the dataset, consequently degrading the performance of LSTM. In particular, the configuration with a batch size of 64 and sequence length of 48 improved the MSE results in up to 90% in respect to other configurations.



(a) MSE

(b) MAE



(c) Correlation

Figure 6.6: Results of the batch size and sequence length for LSTM

We explored the best configuration for the LSTM, considering the assessed datasets with this analysis. Thus, for the following evaluation, the configuration of a batch size of 64 and sequence length of 48 will be used.

6.4.5 Evaluation of Urban Dynamics Prediction: Mobility and Safety

To analyze the performance of the predictions made by the VTq, we compared its performance with the following predictors: *(i)* **RND**: random predictor that selects a value between the minimum value and the maximum value of the value range; *(ii)* **LAST**: predictor that copies the last value of the input data to represent the prediction of the future value; *(iii)* **ARIMA**: integrated autoregressive predictor of moving averages, model adjusted to the time series data to predict future points in the series based on the average of the last input values; *(iv)* **LR**: predictor that uses a linear regression to predict the next value in the time series; *(v)* **CNN**: predictor that uses a convolution neural network to predict future time series data; and *(vi)* **LSTM**: predictor based on recurrent neural

networks implemented by VTq to predict future values of the time series.

The mean square error (MSE), mean absolute error (MAE), and the correlation coefficient were used for the evaluation of the predictors, which are the metrics used for regression analysis. Table 6.1 presents the results of each metric for each predictor, considering mobility and safety data.

Table 6.1: Results of the predictors for the analyzed metrics considering mobility and safety

	Mobility			**Safety**		
Predictors	**MSE**	**MAE**	**Correlation**	**MSE**	**MAE**	**Correlation**
RND	0.1243	0.2899	-3.88	0.1318	0.2988	-3.94
LAST	0.0135	0.0902	0.47	0.0150	0.0931	0.42
ARIMA	0.0112	0.0873	0.53	0.0131	0.0872	0.49
LR	0.0053	0.0510	0.75	0.0075	0.0659	0.71
CNN	0.0055	0.0522	0.75	0.0074	0.0674	0.71
LSTM	**0.0012**	**0.0234**	**0.96**	**0.0019**	**0.0334**	**0.93**

As it can be seen, LSTM implemented by VTq presents better results in all the metrics evaluated for both databases. In particular, LSTM shows a reduction in MSE and MAE by at least 75% and 50%, respectively. LSTM increases the correlation by at least 27% compared to the other solutions. RDN, LAST, and ARIMA solutions cannot make accurate predictions as they do not implement any robust method for making predictions. On the other hand, LR and CNN solutions have better results when compared to RDN and LAST. However, due to the lack of precision in predictions, LR and CNN solutions can introduce false positives and false negatives during route planning, i.e., vehicles can be directed to congested or dangerous roads due to errors in the predictions.

The superior performance presented by LSTM results from its ability to explore the inter-correlation of spatiotemporal data and be able to remember and learn long series of data. The results presented by LSTM show an efficient solution to provide information on future urban dynamics for route planning services because even in wrong predictions, the error introduced by LSTM is very low, consequently not degrading the efficiency of planning. This can be seen in Figure 6.7, which shows the results of the predictions for the safety data comparing all solutions. The y axis shows the values predicted by the solution, while the x axis shows the real values. Therefore, the more scattered the points are, the worse the predictions are, and the greater is the error introduced. It is important to note that mobility results were omitted because the performance of the solutions was similar in both databases.

Figure 6.8 shows the results of the security predictions for a set of 170 predictions from October 2019 for each solution. As it can be seen, the LR and CNN solutions cannot express the dynamics of security adequately. Besides, this behavior makes clear the introduction of false positives and false negatives (see Figures 6.8(d) and 6.8(e) the predicted values are much higher/smaller than the real values). On the other hand, LSTM can express the same dynamics without introducing false positives and false negatives (see Figure 6.8(f)).

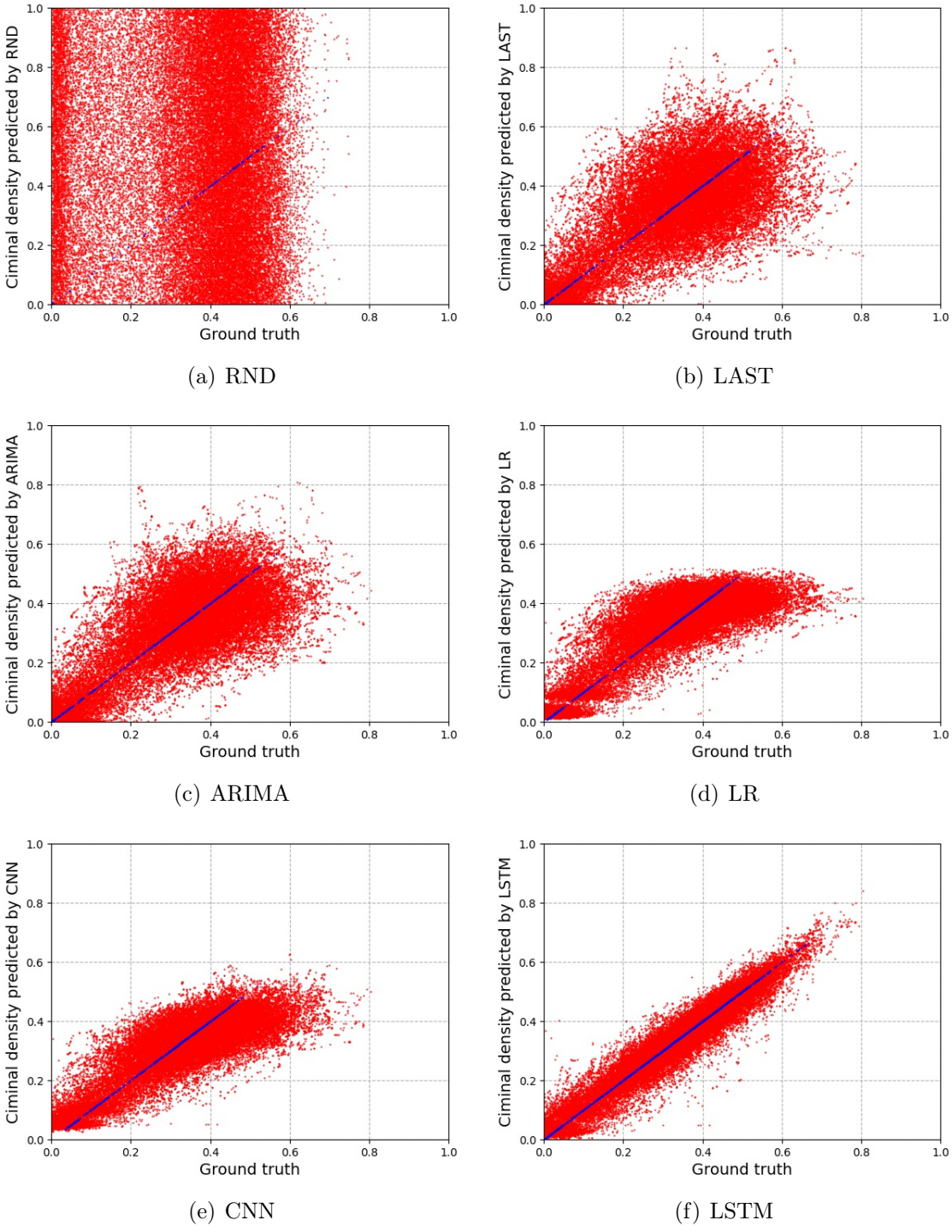
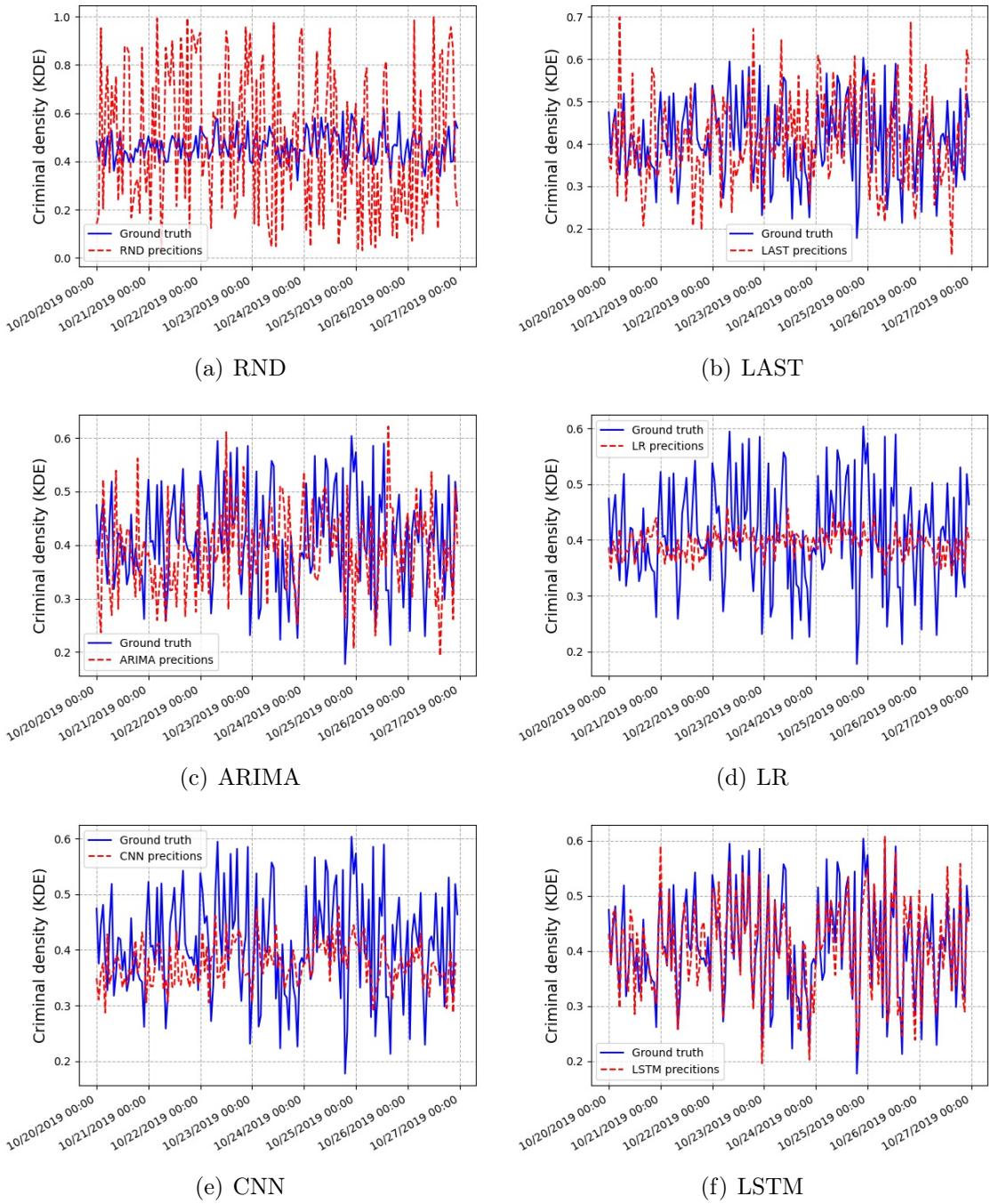


Figure 6.7: Results of the spread of the the predictions: predictions (y axis) vs. ground truth (x axis)

6.4.6 Evaluation of Experience Replay and Exploration Exploitation Trade-off

This subsection evaluates how the ϵ-greedy algorithm can reduce the exploration and exploitation trade-off. Also, we also show how the experience replay technique can speed up the convergence of the proposed algorithm.



(a) RND (b) LAST

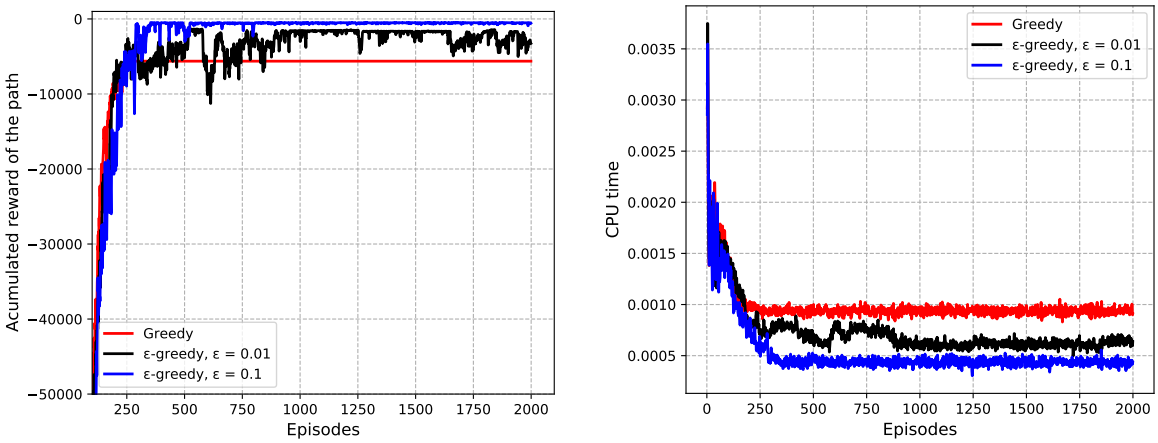
(c) ARIMA (d) LR

(e) CNN (f) LSTM

Figure 6.8: Results of the comparison of the predicted dynamics in respect to the real dynamics of the scenario

Figure 6.9 shows the results for the ϵ-greedy algorithm in function of the episodes to the most efficient path from a state s_0 (e.g., origin of the vehicle) to a state s_t (i.e., destination of the vehicle), considering $\epsilon = 0.01$, $\epsilon = 0.1$, and $\epsilon = 0$ (i.e., greedy), which means that for $\epsilon = 0.01$ the algorithm will explore 1% and exploit 99% of the iterations of the algorithm, consequently, for $\epsilon = 0.1$ the algorithm will explore 10% of the iteration and exploit 90% of the iterations. Thus, the greedy solution (e.i., $\epsilon = 0$) the algorithm will always exploit and never explore.

The results show the importance of the exploration phase to find the most efficient path



(a) Accumulated reward

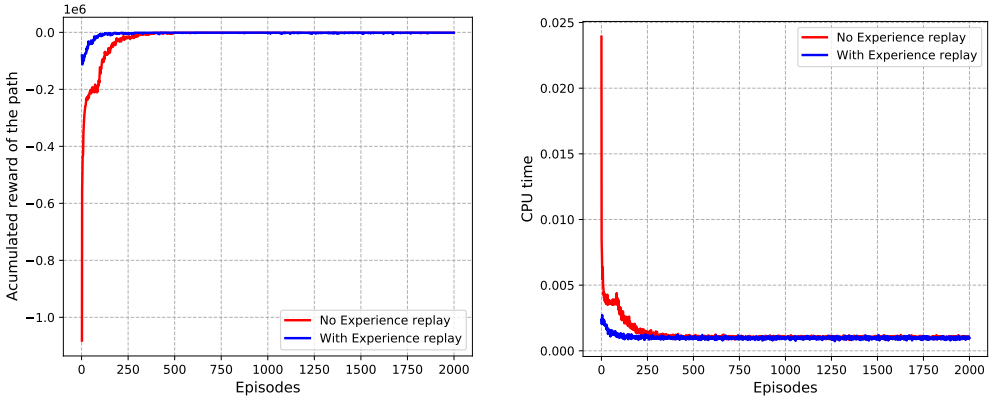
(b) CPU time per episode

Figure 6.9: Results of the exploration and exploitation trade-off

and avoid converging to a local optimum. Thus, as the greedy solution never explores, it could not converge to the global optimum (see Figure 6.9(a)). On the other hand, the ϵ-greedy algorithm with $\epsilon = 0.1$ converged to the global optimum faster than the other variations (approximately 400 episodes). This is a consequence of the exploration phase, enabling the algorithm to gather knowledge about alternative paths and then use such knowledge to extract the most efficient path. However, the value of ϵ needs to be tuned properly, since in some cases, it can lead to a slow convergence as a consequence of the randomness introduced by the exploration phase (see Figure 6.9(a) the ϵ-greedy with $\epsilon = 0.01$). It is worth noticing that the reward values are expressed negatively to avoid cycles during the path planning. Thus if the algorithm plans a route with a cycle, the accumulated reward will be higher than the route planned without the cycle.

Reducing the convergence time of the algorithm is an essential issue to be handled for route planning algorithm, since the longer the time to find the most efficient and reliable path, the longer is the time to re-route the vehicles, consequently introducing an undesired latency to the system which potentially degrades its overall efficiency. Figure 6.9(b) shows the CPU time per episode for each variance of the ϵ-greedy algorithm. The results show a higher CPU time during the exploration phase, but the CPU time remains constant as soon as the algorithm converges. In particular, the ϵ-greedy solution reduces the average CPU time by up to 50% when compared to the greedy solution, which is a consequence of the faster convergence of the algorithm. Despite the good results for the convergence and CPU time, it still can be improved using the experience replay technique.

For the experience replay evaluation, we trained the Q-learning algorithm in a different scenario and then used it in a new scenario. In other words, we saved the Q matrix from a different scenario. Then instead of using an empty matrix to bootstrap the algorithm, we used the Q matrix with the coefficients of the other scenario. This technique is not used to provide the solution for route planning. Instead, it is used to guide the exploration phase towards the vehicle's destination, avoiding the randomness introduced in the early steps of the exploration phase. However, the algorithm still needs to learn the new environment's rewards.



(a) Accumulated reward (b) CPU time per episode

Figure 6.10: Results of experience replay

Figure 6.10 shows the improvement in the convergence and CPU time for the route planning algorithm considering the ϵ-greedy algorithm with $\epsilon = 0.1$. The experience replay reduces the convergence time since it avoids the randomness of the early steps of the exploration phase by guiding the ending state's exploration. In particular, the exploration replay technique improves the convergence time by approximately 40% when compared to the solution without the experience replay (see the episode that the algorithm reaches the constant value in Figure 6.10(a)). Hence, the faster the algorithm converges, the earlier it stops to explore, thus reducing the average CPU time of the algorithm. In particular, the experience replay provides a reduction higher than 60% in the average CPU time per episode (see Figure 6.10(b)).

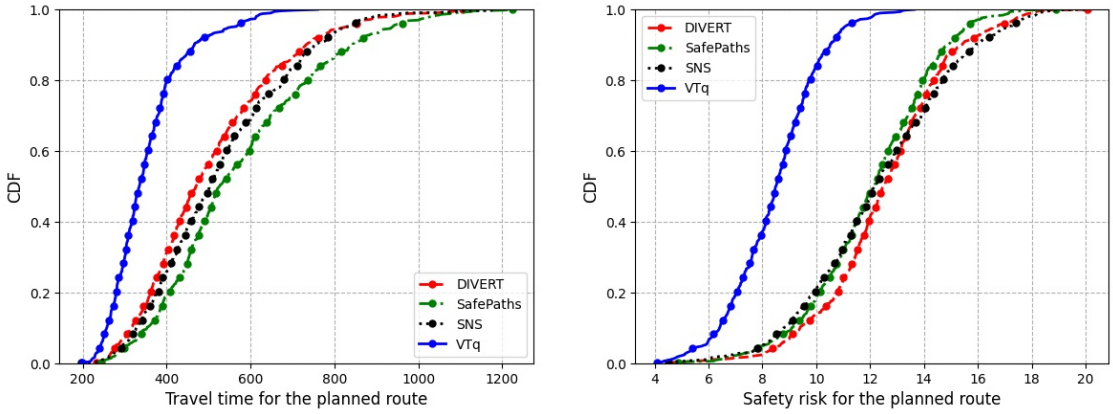
These results have shown the efficiency of the ϵ-greedy algorithm in the exploration and exploitation trade-off and the efficiency of the experience replay to reduce the convergence time of the route planning algorithm and also the CPU time. Thanks to these two mechanisms combined, the proposed route planning algorithm does not introduce an undesired latency to the system. Also, it does not degrade its overall efficiency since it allows real-time decision-making (e.g., it enables route planning in real-time), which is an essential issue in traffic management solutions.

6.4.7 Evaluation of VTq vs Literature Solutions

To evaluate the performance of the VTq, its performance was compared with state-of-the-art solutions for route planning based on mobility and safety, which are: *(i)* DIVERT [54]; *(ii)* SafePaths [34]; and *(iii)* SNS [15]. In this evaluation, travel time and safety risk metrics were used for route planning in addition to the routing algorithm's CPU time.

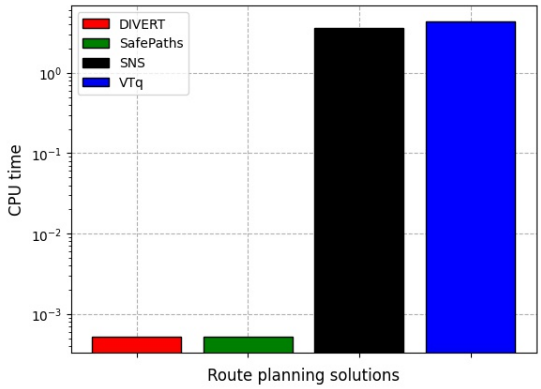
Figure 6.11 shows the comparison of the performance of VTq in relation to the solutions in the literature. In particular, Figure 6.11(a) presents the results of travel time, while Figures 6.11(b) and 6.11(c) present the results of safety risk and CPU time, respectively.

SafePaths plans the route based on the safest route, while DIVERT considers the travel time on the roads, and the SNS explores the trade-off between mobility and safety.



(a) Travel time

(b) Safety risk



(c) CPU time

Figure 6.11: Result of the comparison of the route planning of VTq against literature solutions.

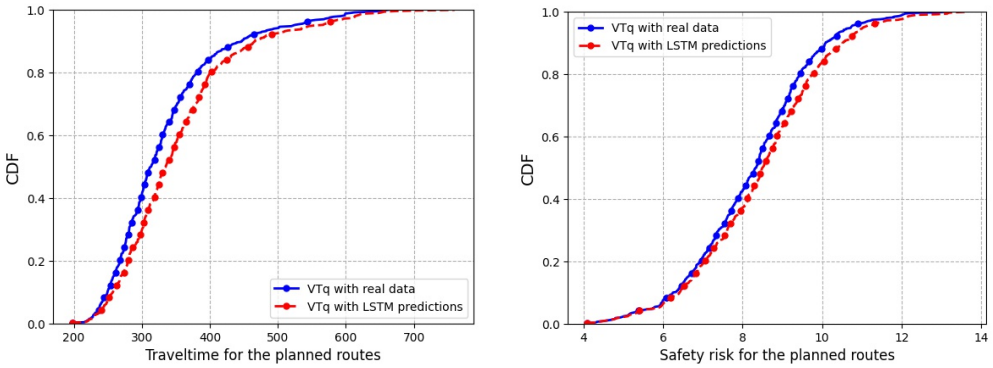
Therefore, DIVERT has the lowest travel time than SNS and SafePaths, and the worst safety results. SafePaths provides the best safety results for drivers and passengers compared to SNS and DIVERT. However, it does not present a good travel time for the users. Finally, SNS is not penalized on any metric for exploiting the trade-off between mobility and security. However, although these solutions were developed to deal with mobility and security problems, none considers the future dynamics of these factors during planning. Consequently, the planned route could potentially cease to be the most efficient shortly, thus degrading the efficiency of planned routes.

The benefits of considering the variation in the future dynamics of urban factors can be seen by comparing the performance of VTq concerning other solutions. Because, even with specific solutions to minimize mobility and security problems, they cannot plan efficient routes due to the dynamics of the urban environment.

In particular, VTq improves the mobility by at least 50% for 90% of the vehicles and improving safety by at least 30% for 90% of the vehicles when compared to solutions for planning routes based on mobility and security that do not consider urban dynamics. Also, VTq has a CPU time equivalent to the SNS solution (see Figure 6.11(c)), which proved to be a solution that does not introduce undesired latency into the system besides being scalable [15]. On the other hand, the DIVERT and SafePaths solutions use less

CPU time than VTq and SNS, which is the result of the shortest path algorithm that has lower complexity when compared to the other solutions. In particular, DIVERT and SafePaths have $O(E + V \log V)$ complexity, while SNS has $O(E \cdot \lambda)$ complexity λ is the maximum risk acceptable for the route planned. Finally, VTq presents a complexity of $O(K \cdot C)$, where K is the number of paths explored and C is the cost to find each path, where C is at least $E + V \log V$.

It is important to clarify that the SafePaths, DIVERT, and SNS solutions can provide the same performance as VTq, but, for this, new planning would have to be done when the vehicle reached the end of each route of its previously planned route. However, this would be potentially impractical as the complexity of the system would increase due to the large number of messages exchanged between server and vehicles (to obtain updated knowledge about the urban dynamics) in addition to reducing the quality of the driving experience, since the driver would have to switch routes very often.



(a) traveled time (b) Safety risk

Figure 6.12: Results of the comparison of the proposed route planning algorithm considering real and predicted data.

Figure 6.12 shows the comparison of the route planning using the values predicted by the LSTM and with the route planning with the actual data. With this evaluation, it is possible to observe the degradation of the route planning due to wrong predictions and identify false positives. However, the results show that VTq has a very similar performance for both plans regardless of its data. Therefore, these results show that the predictions made by LSTM do not degrade the efficiency of the route planning employed by the VTq, nor do they introduce false positives and false negatives in this process.

6.5 Chapter Conclusions

This chapter presented VTq, an efficient route planning system that predicts future urban dynamics and considers future changes during route planning to provide more efficient and reliable routes. To predict future changes in urban dynamics, VTq implements an LSTM, which provides accurate predictions about future urban dynamics by exploring the temporal correlation in the historical data. On the other hand, to consider the predicted future changes in the urban environment during the route planning, VTq implements a

reinforcement learning-based algorithm, which learns how to compute the most efficient route through a set of trial-and-error approaches by iterating through the environment considering the predicted changes.

One important issue that needs attention when considering route planning systems is the computation time. In this way, to enable real-time re-routing, VTq applies two techniques to improve the convergence time of the route planning algorithm, named: *(i)* ϵ-greedy algorithm to improve the exploration and exploitation trade-off; and *(ii)* the experience replay technique that uses the knowledge from different scenarios to improve the search for the most efficient route. By using these mechanisms, VTq provides a suitable routing algorithm with computation time equivalent to SNS, which showed to be a suitable solution for traffic management problems [15].

The results have shown substantial improvements in the planned routes provided by VTq when compared to state-of-the-art solutions for traffic management considering mobility and safety. However, VTq uses the Q-learning algorithm to perform the reinforcement learning methods, which presents storage and computation limitations depending on the scenario since the algorithm needs to store the reward matrices and Q-values (which are a graph representation of the environment). However, the rewards and Q-values storage depends not only on the environment size but also on the prediction window used by the LSTM. Thus, depending on the prediction window and the scenario size, the storage capabilities and computation time might become impractical for real-time systems. Thus, an efficient deep Q-learning algorithm can eliminate the memory problems of the conventional Q-learning algorithm by using a neural network to make the decisions instead of using tabular values, consequently handling such a problem. This will be treated as future work of this thesis. Also, the research opportunities provided by this thesis are presented in the next chapter.

Chapter 7

Final Remarks

This chapter presents the conclusion for the solutions presented in this work in a top-down approach. Moreover, Section 7.2 describes the potential future work and also research opportunities provided by the solutions presented in this thesis. At last, Section 7.3 presents the publications achieved by this work and collaborations.

7.1 Thesis conclusion: A Top-down Approach

Chapter 1 introduced the mobility problems and how a traffic management system should be designed to overcome those issues. In addition, the main issues that need to be handled to propose an efficient traffic management system were presented, including: *(i)* network overhead; *(ii)* computational efforts; *(iii)* accurate knowledge about urban dynamics; and *(iv)* efficient multi-objective and personalized re-routing. In this way, motivated by those issues, the chapter introduced the main research question that the thesis aims to answer, listed as *(i)* how to ensure the system scalability with low overhead and still with good traffic management? *(ii)* how to enable an efficient and personalized multi-objective re-routing without creating additional congestion spots, and *(iii)* how to consider future changes in urban dynamics during the re-routing to plan a more efficient and reliable route?

Chapter 2 presented the main technologies used by this thesis. First, a brief description of traffic management systems was presented and showed how the sensing, communication, and processing technologies could cope together to enable a more efficient and reliable TMS. In this way, a list of potential communication technologies that will pave the way for TMS services and applications was described, including: *(i)* vehicle-to-everything; *(ii)* visible light communication; *(iii)* 5G networks; and *(iv)* Multi-access edge computing. Another technology that will empower TMS services is machine learning-based solutions. Thus, a description of how machine learning methods and algorithms can improve TMS applications was shown.

Chapter 3 described literature solutions for the traffic management problem, highlighting advantages and limitations for each solution. Also, a classification and a qualitative comparison were proposed. In summary, the literature solutions present have limitations related to: *(i)* system scalability; *(ii)* multi-objective and personalized re-routing; and

(iii) lack of knowledge about future changes in the urban dynamics.

Chapter 4 introduced SIC, a cooperative routing algorithm for improving traffic efficiency. SIC was design based on two major principles for vehicular traffic management: *(i)* real-time vehicular traffic re-routing; and *(ii)* network contention minimization. To do so, it offloads the route computation in each vehicle, reducing the computation time and the communication burden on the server, consequently providing better scalability to the system. Moreover, it employs a cooperative re-routing algorithm, in which the vehicles are aware of the routes taken by their neighbors, thus, providing better traffic management. Results showed that SIC provides a suitable architecture for traffic re-routing, which produces a low overhead and low complexity and CPU time (which enables a real-time system). Hence, enabling a highly scalable system and cooperative re-routing algorithm.

However, SIC only considers the traffic condition on the scenario to re-route vehicles, which is far from the desired requirements for future TMS, because several different urban aspects can be taken into consideration during route planning decisions, such as distance, fuel consumption, CO_2 emissions, scenery, and even safety risks. All vehicles are re-routed according to the same criteria, but different users may have different preferences according to their path planning decisions. For instance, cautious users may prefer a safer but longer route, while users with limited time tend to prefer faster routes [15]. Therefore, this thesis will address the issues related to multi-objective and personalized re-routing for traffic management systems considering different urban aspects and their spatiotemporal correlation in the next chapter.

Chapter 5 have presented SNS, a non-deterministic multi-objective re-routing system for improving traffic efficiency while improving the safety of drivers and passengers. To do so, SNS extracts knowledge about traffic conditions and safety risks over the city. On the one hand, the knowledge about traffic conditions is obtained based on the vehicles' traffic information. On the other hand, to extract the knowledge about safety risks over the city, SNS proposes an algorithm to discover the safety risks based on criminal activities. The key idea of the algorithm is to identify hotspots for criminal activities using a KDE for each type of crime. In this way, by segmenting the set of crimes that happened throughout the day considering a time window of one hour, SNS explores the spatiotemporal correlation, which means that the system can identify not only the regions that are more likely to provide higher chances for criminal activities but also when those regions will become dangerous.

With both pieces of knowledge, SNS employs a non-deterministic multi-objective re-routing to improve the traffic efficiency and the safety of drivers and passengers. The re-routing algorithm reduces the problem of creating additional congestion spots by distributing the traffic flow over the routes in the Pareto-set (i.e., routes that improve both metrics, mobility, and safety). Besides, SNS explores the spatiotemporal correlation of criminal activities and enables personalized re-routing (in which the drivers can choose which type of crimes they want to avoid) to push the system's efficiency further.

Yet, SNS does not care about future changes in the urban dynamics, which means that future changes in either the traffic efficiency or the safety dynamics might degrade the efficiency and the reliability of the routes previously computed. To minimize such a problem, the system relies on periodic re-routing. However, this procedure increases the

network usage and the computational efforts of the system because the system needs to gather the traffic information over the scenario and compute new routes to each vehicle during each re-routing phase. Knowing when the changes in the urban dynamics will happen beforehand, it is essential to recommend more reliable and efficient routes.

Chapter 6 have presented VTq, an efficient route planning system that predicts future urban dynamics and considers future changes during the route planning to provide more efficient and reliable routes. To predict future changes in urban dynamics, VTq implements an LSTM, which provides accurate predictions about future urban dynamics by exploring the temporal correlation in the historical data. On the other hand, to consider the predicted future changes in the urban environment during the route planning, VTq implements a reinforcement learning-based algorithm that learns how to compute the most efficient route through a set of trial-and-error approaches by iterating through the environment considering the predicted changes.

One important issue that needs attention when considering route planning systems is the computation time. In this way, to enable real-time re-routing, VTq applies two techniques to improve the convergence time of the route planning algorithm, named: *(i)* ϵ-greedy algorithm to improve the exploration and exploitation trade-off; and *(ii)* the experience replay technique that uses the knowledge from different scenarios to improve the search for the most efficient route. By using these mechanisms, VTq provided a suitable routing algorithm with computation time equivalent to SNS, which showed to be a suitable solution for traffic management problems [15].

The results have shown substantial improvements in the planned routes provided by VTq when compared to state-of-the-art solutions for traffic management considering mobility and safety. However, VTq uses the Q-learning algorithm to perform the reinforcement learning methods, which presents storage and computation limitations depending on the scenario since the algorithm needs to store the reward matrices and Q-values (which are a graph representation of the environment). However, the rewards and Q-values storage depends not only on the environment size but also on the prediction window used by the LSTM. Thus, depending on the prediction window and the scenario size, the storage capabilities and computation time might become impractical for real-time systems. Thus, an efficient deep Q-learning algorithm can eliminate the memory problems of the conventional Q-learning algorithm by using a neural network to make the decisions instead of using tabular values, consequently handling such a problem. This will be treated as future work of this thesis. Also, the research opportunities provided by this thesis are presented in the next chapter.

Finally, Chapter 7 concludes the thesis presenting the publications achieved, the future work and research opportunities for the issues described in this thesis.

7.2 Open Challenges and Future Work

This section provides a brief description of potential future work and research opportunities for the solutions presented in this thesis.

7.2.1 Heterogeneous Data Integration

The main challenge is how to integrate data from different sources since we have many different systems and sources without integration among them, providing a massive amount of data with no standardization. Furthermore, as emerging technologies such as IoT will provide data exchange and communication to many everyday life devices, it is essential to use these devices to turn the data collection paradigm into a new one. However, with this integration, many other challenges will arise, including tracking and managing the high number of devices involved in such integration. Current open problems are: how to define novel approaches for device identification and the generation of unique identifiers; how to use these identifiers as addresses to forward and route information; How to employ an IoT-based identifier for TMS. At last, the information collected from these devices may carry private information about the owners, and as these transmissions may suffer attacks, a secure mechanism to protect this information is desired.

7.2.2 Data Management and Big Data Issues

TMSs need to handle a huge amount of data. Therefore, standardization in data representation needs to be employed. Once many problems may arise, each source employs independent measurements and formatting. Moreover, many sources may report their data asynchronously. Thus, a big challenge is how to manage such an issue.

Furthermore, data correlation is another challenge due to non-integration among different systems and sources, in which the same source may provide data in different systems. In other words, as different systems are independent, data accounting can result in false positives. However, the challenge is to correlate such data to a common source. Besides, TMSs need to provide sophisticated mechanisms to fuse, aggregate, and exploit data to deal with different data types provided from heterogeneous sources. The major challenge is how to exploit these big data issues in a vehicular environment, once that the current models and algorithms used in big data are physically and logically decentralized but virtually centralized [31].

7.2.3 Alternative Route Guidance

Suggesting and computing alternative routes to avoid traffic hazards is the best way to improve overall traffic efficiency. However, the main challenge is how to do this appropriately to avoid an undesired overhead, consequently avoiding the vehicles getting stuck in some congestion. Although relying on central entities (centralized approach) to compute and suggest alternative routes to all vehicles is more efficient due to its better management and scenario overview, but depending on the number of vehicles to be re-routed and the complexity of the algorithm used in the alternative route computation, such approach may introduce high overhead degrading its performance. With this problem in mind, one solution is to enable each vehicle to compute its alternative route. However, the key challenge is how to provide a full scenario overview about the traffic condition to every vehicle to enable them to compute an efficient route without overloading the network. Another concern is how to compute an efficient alternative route without resulting in

traffic congestion in other areas shortly, providing a better traffic balance and management. In this way, to have good alternative route guidance, a trade-off between efficiency and complexity is essential.

7.2.4 Accurate Vehicle Localization

Precise localization is an essential task for autonomous driving which will be the backbone of the TMS applications; it is necessary to enable more reliable decision-making, effective collision avoidance, and path planning. In these scenarios, vehicles need to localize themselves, other vehicles, and obstacles in their surroundings with errors on a centimeter scale to avoid potential damages and enable trusted guidance. In this sense, autonomous driving services must rely on alternative localization approaches such as LIDAR information and environmental sensors to achieve desirable localization performance.

7.2.5 Intelligent Traffic Management

The reinforcement learning approach for path planning has led to substantial improvements in the planning of efficient routes. However, it can still decrease overall traffic efficiency due to its deterministic feature, allowing vehicles with a similar origin and destination to plan the same path under the same urban dynamics. To overcome this issue, cooperative approaches must be integrated during path planning. Therefore, before planning its path, a vehicle should know the paths already taken by others to avoid planning the same path.

7.2.6 Integration with Non-autonomous Vehicles

Before achieving a fully-autonomous transportation system, autonomous vehicles will share roads and urban environments with non-autonomous ones. In this scenario, autonomous vehicles should interact with non-autonomous ones to avoid potential accidents and keep traffic mobility. However, in such challenging integration, autonomous vehicles need to: (i) understand drivers' behaviors; (ii) detect and predict their movements; (iii) send alerts and notification; and (iv) estimate trajectories.

7.2.7 Humans and Autonomous Driving

Autonomous vehicles will share roads with vulnerable road users (VRUs), such as pedestrians and cyclists, and interaction with the TMS will be required to enable safe and smooth driving. These interactions can be direct and indirect. Direct interactions enable VRUs to provide their intentions to vehicles using V2X communications or body gestures. In contrast, indirect interactions can be acquired through environment sensors and cameras to identify distracted VRUs or even predict their movements to avoid accidents. However, TMS also needs to pay attention to passengers, deciding when they should control some situations (i.e., construction zones, police-controlled intersections, crucial moments). In these cases, TMS needs to know whether the passenger can take over control to trigger such a task correctly. In other words, are they paying attention? Did they engage in

another secondary activity? Are they hands-free? Have they been alerted to change to the driving environment?

7.2.8 Vehicle Security

In addition to common issues related to data security and privacy present in vehicular networks and intelligent transportation services, TMS applications must care about physical security since malicious attackers could take total control over the vehicle, enabling them to perform malicious actions and put the life of VRUs and passengers at risk. These malicious actions can, for instance, cause accidents on purpose and detour vehicles of their routes. Thus, vehicle security must be a core component in autonomous vehicles, which needs to develop methods to ensure a highly secure environment.

7.3 Publications

7.3.1 Work Published in Journals

1. [15] **Souza, A. M.**, Braun, T., Botega, L., Villas, L. A., and Loureiro, A. F., *Safe and Sound: Driver Safety-aware Vehicle Re-routing based on Spatiotemporal Information.* IEEE Transactions on Intelligent Transportation Systems (T-ITS). 2020

2. [18] **Souza, A. M.**, Oliveira, H. F., Zhao, Z., Braun, T., Villas, L. A., and Loureiro, A. F. *Enhancing Sensing and Decision-Making of Automated Driving Systems with Multi-access Edge Computing and Machine Learning.* IEEE Intelligent Transportation System Magazine (ITS-Magazine). 2020

3. [28] **Souza, A. M.**, Maia, G., Braun, T., and Villas, L. A., *An Interest-based Approach for Reducing Network Contentions in Vehicular Transportation Systems.* MDPI Sensors. 2019

4. [26] **Souza, A. M.**, Braun, T., Botega, L., Cabral, R., Garcia, I., and Villas, L. A., *Better Safe Than Sorry: A Vehicular Traffic Re-routing based on Traffic Conditions and Public Safety Issues.* Springer Journal of Internet Services and Applications (JISA). 2019

5. [27] **Souza, A. M.**, Brennand C. A., Yokoyama R. S., Donato E. A., Madeira E. R., Villas L. A. *Traffic management systems: A classification, review, challenges, and future perspectives.* International Journal of Distributed Sensor Networks. 2017

7.3.2 Work Published in Conferences

1. [30] **Souza, A. M.**, and Villas L. A., *A Fully-distributed Traffic Management System to Improve Overall Traffic Efficiency.* Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM), 2017.

2. [17] **Souza, A. M.**, Fonseca, N. L., and Villas, L. A. *A Fully-distributed Advanced Traffic Management System based on Opportunistic Content Sharing.* 2017 IEEE International Conference on Communications (ICC), 2017

3. [25] **Souza, A. M.**, Botega, L., and Villas, L. A., *GTE: Um Sistema para Gerenciamento de Trânsito Escalável baseado em Compartilhamento Oportunista.* XXXV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2017. **Honorable mention**.

4. [24] **Souza, A. M.**, Botega, L., Garcia, I. C., and Villas, L. A., *Por Aqui é Mais Seguro: Melhorando a Mobilidade e a Segurança nas Vias Urbanas.* XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2018. **Honorable mention**.

5. [23] **Souza, A. M.**, and Villas, L. A., *Vem Tranquilo: Rotas Eficientes baseado na Dinâmica Urbana Futura com Deep Learning e Computação de Borda .* XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2020.

6. [19] **Souza, A. M.**, Pedrosa, L. C. L., Botega, L., and Villas, L. A., *itsSAFE: An Intelligent Transportation System for Improving Safety and Traffic Efficiency.* IEEE 87th Vehicular Technology Conference (VTC Spring) 2018.

7. [14] **Souza, A. M.**, Botega, L., and Villas, L. A., *FnS: Enhancing Traffic Mobility and Public Safety based on a Hybrid Transportation System.* 14th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), 2018.

8. [16] **Souza, A. M.**, Braun, T., and Villas, L. A., *Efficient Context-Aware Vehicular Traffic Re-Routing Based on Pareto-Optimality: A Safe-Fast Use Case.* 21st International Conference on Intelligent Transportation Systems (ITSC), 2018.

7.3.3 Collaborations

1. Gomides, S. J., **Souza, A. M.**, Souza, S. H. F., Villas, L. A., and Guidoni L. D., *An adaptive and Distributed Traffic Management System using Vehicular Ad-hoc Networks.* Elsevier Computer Communications (ComCom), 2020.

2. Nobre, J. C., **Souza, A. M.**, Rosário, D., Both, C., Villas, L. A., Cerqueira, E., Braun, T., and Gerla, M., *Vehicular software-defined networking and fog computing: integration and design principles.* Elsevier Ad Hoc Networks, 2019.

3. Costa, J., **Souza, A. M.**, Rosário, D., Villas, L. A., and Cerqueira, E., *Efficient data dissemination protocol based on complex networks' metrics for urban vehicular networks.* Springer Journal of Internet Services and Applications (JISA), 2019.

4. Ladeira, L. Z., **Souza, A. M.**, Pereira, G. R. F., Silva, T. H., and Villas, L. A., *Serviço de Sugestão de Rotas Seguras para Veículos.* XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2019.

5. Costa, J., **Souza, A. M.**, Lobato, W., Rosário, D., Villas, L. A., and Cerqueira, E., *Centrality-based data dissemination protocol for vehicular ad hoc networks.* IEEE 16th International Symposium on Network Computing and Applications (NCA), 2017.

6. Costa, J., **Souza, A. M.**, Rosário, D., Villas, L. A., and Cerqueira, E., *Protocolo para Disseminação de Dados em VANETs baseado em Métricas de Redes Complexas: Um Estudo de Caso com Sistema de Gerenciamento de Trânsito.* XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2018.

7. Costa, J., **Souza, A. M.**, Rosário, D., Villas, L. A., and Cerqueira, E., *Data Dissemination Based on Complex Networks' Metrics for Distributed Traffic Management Systems.* IEEE Symposium on Computers and Communications (ISCC), 2017.

Bibliography

[1] A. Ahmad, S. Din, A. Paul, G. Jeon, M. Aloqaily, and M. Ahmad. Real-time route planning and data dissemination for urban scenarios using the internet of things. *IEEE Wireless Communications*, 26(6):50–55, 2019.

[2] Ademar T. Akabane, Roger Immich, Luiz F. Bittencourt, Edmundo R.M. Madeira, and Leandro A. Villas. Towards a distributed and infrastructure-less vehicular traffic management system. *Computer Communications*, 151:306–319, 2020.

[3] F. Altché, X. Qian, and A. de La Fortelle. An algorithm for supervised driving of cooperative semi-autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3527–3539, Dec 2017.

[4] Mahyar Amirgholy, Nima Golshani, Craig Schneider, Eric J. Gonzales, and H. Oliver Gao. An advanced traveler navigation system adapted to route choice preferences of the individual users. *International Journal of Transportation Science and Technology*, 6(4):240 – 254, 2017. Special Issue on Urban Spatiotemporal Behavior and Network Assignment.

[5] M. Ayyash, H. Elgala, A. Khreishah, V. Jungnickel, T. Little, S. Shao, M. Rahaim, D. Schulz, J. Hilt, and R. Freund. Coexistence of wifi and lifi toward 5g: concepts, opportunities, and challenges. *IEEE Communications Magazine*, 54(2):64–71, February 2016.

[6] J. H. Banks. *Introduction to transportation engineering*. McGraw, New York, 2002.

[7] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. SUMO - Simulation of Urban MObility: An Overview. In *International Conference on Advances in System Simulation (SIMUL '11)*, pages 63–68, 2011.

[8] C. Brennand, A. Souza, Guilherme Maia, A. Boukerche, H. S. Ramos, A. A. F. Loureiro, and Leandro A. Villas. An intelligent transportation system for detection and control of congested roads in urban centers. In *IEEE Symposium on Computers and Communication (ISCC '15)*, pages 1–6, 2015.

[9] C. A. R. L. Brennand, F. D. da Cunha, G. Maia, E. Cerqueira, A. A. F. Loureiro, and L. A. Villas. Fox: A traffic management system of computer-based vehicles fog. In *2016 IEEE Symposium on Computers and Communication (ISCC)*, pages 982–987, June 2016.

[10] Celso ARL Brennand, Allan M de Souza, Guilherme Maia, Azzedine Boukerche, Heitor Ramos, Antonio AF Loureiro, and Leandro A Villas. An intelligent transportation system for detection and control of congested roads in urban centers. In *Computers and Communication (ISCC), 20th IEEE Symposium on*, pages 476–481. IEEE, 2015.

[11] C. Campolo, A. Molinaro, A. Iera, and F. Menichella. 5g network slicing for vehicle-to-everything services. *IEEE Wireless Communications*, 24(6):38–45, Dec 2017.

[12] Cable News Network (CNN). Waze app directions take woman to wrong brazil address, where she is killed, 2015.

[13] A. Căilean and M. Dimian. Current challenges for visible light communications usage in vehicle applications: A survey. *IEEE Communications Surveys Tutorials*, 19(4):2681–2703, Fourthquarter 2017.

[14] A. M. de Souza, L. C. Botega, and L. A. Villas. Fns: Enhancing traffic mobility and public safety based on a hybrid transportation system. In *2018 14th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 77–84, 2018.

[15] A. M. de Souza, T. Braun, L. C. Botega, L. A. Villas, and A. A. F. Loureiro. Safe and sound: Driver safety-aware vehicle re-routing based on spatiotemporal information. *IEEE Transactions on Intelligent Transportation Systems*, 21(9):3973–3989, 2020.

[16] A. M. De Souza, T. Braun, and L. Villas. Efficient context-aware vehicular traffic re-routing based on pareto-optimality: A safe-fast use case. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2905–2910, Nov 2018.

[17] A. M. de Souza, N. L. S. da Fonseca, and L. Villas. A fully-distributed advanced traffic management system based on opportunistic content sharing. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, May 2017.

[18] A. M. de Souza, H. F. Oliveira, Z. Zhao, T. Braun, L. Villas, and A. A. F. Loureiro. Enhancing sensing and decision-making of automated driving systems with multi-access edge computing and machine learning. *IEEE Intelligent Transportation Systems Magazine*, pages 0–0, 2020.

[19] A. M. de Souza, L. L. C. Pedrosa, L. C. Botega, and L. Villas. Itssafe: An intelligent transportation system for improving safety and traffic efficiency. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–7, June 2018.

[20] A. M. de Souza, L. L. C. Pedrosa, L. C. Botega, and L. Villas. Itssafe: An intelligent transportation system for improving safety and traffic efficiency. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–7, June 2018.

[21] A. M. de Souza, R. S. Yokoyama, L. C. Botega, R. I. Meneguette, and L. A. Villas. Scorpion: A solution using cooperative rerouting to prevent congestion and improve traffic condition. In *Computer and Information Technology, 2015 IEEE International Conference on*, pages 497–503, Oct 2015.

[22] A. M. de Souza, R. S. Yokoyama, G. Maia, A. Loureiro, and L. Villas. Real-time path planning to prevent traffic jam through an intelligent transportation system. In *2016 IEEE Symposium on Computers and Communication (ISCC)*, pages 726–731, June 2016.

[23] Allan de Souza and Leandro Villas. Vem tranquilo: Rotas eficientes baseado na dinâmica urbana futura com deep learning e computação de borda. In *Anais do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 351–364, Porto Alegre, RS, Brasil, 2020. SBC.

[24] Allan M. de Souza, Leonardo C. Botega, Islene Calciolari Garcia, and Leandro A. Villas. Por aqui é mais seguro: Melhorando a mobilidade e a segurança nas vias urbanas. *Anais do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, 36, 2018.

[25] Allan M. de Souza, Leonardo C. Botega, and Leandro A. Villas. Gte: Um sistema para gerenciamento de trânsito escalável baseado em compartilhamento oportunista. In *Anais do XXXV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, Porto Alegre, RS, Brasil, 2017. SBC.

[26] Allan M. de Souza, Torsten Braun, Leonardo C. Botega, Raquel Cabral, Islene C. Garcia, and Leandro A. Villas. Better safe than sorry: a vehicular traffic re-routing based on traffic conditions and public safety issues transportation systems. *Springer Journal of Internet Services and Applications*, 10(1), 2019.

[27] Allan M de Souza, Celso ARL Brennand, Roberto S Yokoyama, Erick A Donato, Edmundo RM Madeira, and Leandro A Villas. Traffic management systems: A classification, review, challenges, and future perspectives. *International Journal of Distributed Sensor Networks*, 13(4):1550147716683612, 2017.

[28] Allan M. de Souza, Guilherme Maia, Torsten Braun, and Leandro A. Villas. An interest-based approach for reducing network contentions in vehicular transportation systems. *Sensors*, 19(10), 2019.

[29] Allan M. de Souza, R.S. Yokoyama, Azzedine Boukerche, Guilherme Maia, Eduardo Cerqueira, Antonio A.F. Loureiro, and Leandro Aparecido Villas. Icarus: Improvement of traffic condition through an alerting and re-routing system. *Computer Networks*, 110:118 – 132, 2016.

[30] Allan Mariano de Souza and Leandro Aparecido Villas. A Fully-distributed Traffic Management System to Improve the Overall Traffic Efficiency. In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of*

Wireless and Mobile Systems, MSWiM '16, pages 19–26, New York, NY, USA, 2016. ACM.

[31] S. Djahel, R. Doolan, G.-M. Muntean, and J. Murphy. A Communications-Oriented Perspective on Traffic Management Systems for Smart Cities: Challenges and Innovative Approaches. *IEEE Communications Surveys Tutorials*, 17(1):125–151, 2015.

[32] P. Dong, T. Zheng, S. Yu, H. Zhang, and X. Yan. Enhancing vehicular communication using 5g-enabled smart collaborative networking. *IEEE Wireless Communications*, 24(6):72–79, Dec 2017.

[33] R. Doolan and G. M. Muntean. Ecotrec: A novel vanet-based approach to reducing vehicle emissions. *IEEE Transactions on Intelligent Transportation Systems*, 18(3):608–620, March 2017.

[34] Esther Galbrun, Konstantinos Pelechrinis, and Evimaria Terzi. Urban navigation beyond shortest route. *Inf. Syst.*, 57(C):160–171, April 2016.

[35] Thiago S. Gomides, Robson E. De Grande, Allan M. de Souza, Fernanda S.H. Souza, Leandro A. Villas, and Daniel L. Guidoni. An adaptive and distributed traffic management system using vehicular ad-hoc networks. *Computer Communications*, 159:317–330, 2020.

[36] Mordechai (Muki) Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18, October 2008.

[37] S. Hayes, S. Wang, and S. Djahel. Personalized road networks routing with road safety consideration: A case study in manchester. In *2020 IEEE International Smart Cities Conference (ISC2)*, pages 1–6, 2020.

[38] A. T. Hussein, M. T. Alresheedi, and J. M. H. Elmirghani. 20 gb/s mobile indoor visible light communication system employing beam steering and computer generated holograms. *Journal of Lightwave Technology*, 33(24):5242–5260, Dec 2015.

[39] H.C Joksch. The shortest route problem with constraints. *Journal of Mathematical Analysis and Applications*, 14(2):191 – 197, 1966.

[40] S. Kim and W. Liu. Cooperative autonomous driving: A mirror neuron inspired intention awareness and cooperative perception approach. *IEEE Intelligent Transportation Systems Magazine*, 8(3):23–32, Fall 2016.

[41] Scott Kirkpatrick, C Daniel Gelatt, Mario P Vecchi, et al. Optimization by simmulated annealing. *science*, 220(4598):671–680, 1983.

[42] L. Z. Ladeira, A. M. de Souza, T. H. Silva, R. W. Pazzi, and L. A. Villas. Ponche: Personalized and context-aware vehicle rerouting service. In *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*, pages 211–218, 2020.

[43] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[44] Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. Many-objective evolutionary algorithms: A survey. *ACM Comput. Surv.*, 48(1):13:1–13:35, September 2015.

[45] L. Li, K. Ota, and M. Dong. Humanlike driving: Empirical decision-making system for autonomous vehicles. *IEEE Transactions on Vehicular Technology*, 67(8):6814–6823, Aug 2018.

[46] J. Liu, J. Wan, D. Jia, B. Zeng, D. Li, C. Hsu, and H. Chen. High-efficiency urban traffic management in context-aware computing and 5g communication. *IEEE Communications Magazine*, 55(1):34–40, January 2017.

[47] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E. Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11 – 26, 2017.

[48] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys Tutorials*, 19(4):2322–2358, Fourthquarter 2017.

[49] R. Timothy Marler and Jasbir S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41(6):853–862, Jun 2010.

[50] R.T. Marler and J.S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, Apr 2004.

[51] J. Morton, T. A. Wheeler, and M. J. Kochenderfer. Analysis of recurrent neural networks for probabilistic modeling of driver behavior. *IEEE Transactions on Intelligent Transportation Systems*, 18(5):1289–1298, May 2017.

[52] Mail Online. Dash cam captures the terrifying moment waze smartphone app directs a driver into a gunfight in boston, 2016.

[53] J. Pan, M.A. Khan, I. Sandu Popa, K. Zeitouni, and C. Borcea. Proactive vehicle re-routing strategies for congestion avoidance. In *Distributed Computing in Sensor Systems (DCOSS), 2012 IEEE 8th International Conference on*, pages 265–272, May 2012.

[54] J. Pan, I. Sandu Popa, and C. Borcea. Divert: A distributed vehicular traffic re-routing system for congestion avoidance. *IEEE Transactions on Mobile Computing*, PP(99):1–1, 2016.

[55] L. G. Papaleondiou and M. D. Dikaiakos. Trafficmodeler: A graphical tool for programming microscopic traffic simulators through high-level abstractions. In *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, pages 1–5, April 2009.

[56] Pedro Perez-Murueta, Alfonso Gómez-Espinosa, Cesar Cardenas, and Miguel Gonzalez-Mendoza. Deep learning system for vehicular re-routing and congestion avoidance. *Applied Sciences*, 9(13), 2019.

[57] C. Qiu, Y. Zhang, Z. Feng, P. Zhang, and S. Cui. Spatio-temporal wireless traffic prediction with recurrent neural network. *IEEE Wireless Communications Letters*, 7(4):554–557, Aug 2018.

[58] M. Rezaei, H. Noori, M. Mohammadkhani Razlighi, and M. Nickray. Refocus+: Multi-layers real-time intelligent route guidance system with congestion detection and avoidance. *IEEE Transactions on Intelligent Transportation Systems*, 22(1):50–63, 2021.

[59] D. W. Scott. *Multivariate density estimation: theory, pratice and visualization*. John Wiley Sons, New York, Chicester, 1992.

[60] Sumit Shah, Fenye Bao, Chang-Tien Lu, and Ing-Ray Chen. Crowdsafe: Crowd sourcing of crime incidents and safe routing on mobile devices. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '11, pages 521–524, New York, NY, USA, 2011. ACM.

[61] C. Sommer, R. German, and F. Dressler. Bidirectionally coupled network and road traffic simulation for improved ivc analysis. *Mobile Computing, IEEE Transactions on*, 10(1):3–15, Jan 2011.

[62] A. Taha and N. AbuAli. Route planning considerations for autonomous vehicles. *IEEE Communications Magazine*, 56(10):78–84, OCTOBER 2018.

[63] András Varga and Rudolf Hornig. An overview of the OMNeT++ simulation environment. In *International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (Simutools '08)*, pages 1–10, 2008.

[64] M. Wang, H. Shan, R. Lu, R. Zhang, X. Shen, and F. Bai. Real-Time Path Planning Based on Hybrid-VANET-Enhanced Transportation System. *IEEE Transactions on Vehicular Technology*, 64(5):1664–1678, 2015.

[65] S. Wang, S. Djahel, and J. McManis. An adaptive and vanets-based next road re-routing system for unexpected urban traffic congestion avoidance. In *Vehicular Networking Conference (VNC), 2015 IEEE*, pages 196–203, Dec 2015.

[66] S. Wang, S. Djahel, Z. Zhang, and J. McManis. Next Road Rerouting: A Multiagent System for Mitigating Unexpected Urban Traffic Congestion. *IEEE Transactions on Intelligent Transportation Systems*, 17(10):2888–2899, 2016.

[67] H. Ye, L. Liang, G. Y. Li, J. Kim, L. Lu, and M. Wu. Machine learning for vehicular networks: Recent advances and application examples. *IEEE Vehicular Technology Magazine*, 13(2):94–101, June 2018.

[68] Y. Jin Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.

[69] Weidong Zhang, Nyothiri Aung, Sahraoui Dhelim, and Yibo Ai. DIFTOS: A Distributed Infrastructure-Free Traffic Optimization System Based on Vehicular Ad Hoc Networks for Urban Environments. *SENSORS*, 18(8), aug 2018.

[70] K. Zheng, Q. Zheng, P. Chatzimisios, W. Xiang, and Y. Zhou. Heterogeneous vehicular networking: A survey on architecture, challenges, and solutions. *IEEE Communications Surveys Tutorials*, 17(4):2377–2396, Fourthquarter 2015.