A Multi-Objective Route Planning Framework for Automated and Connected Vehicles

vorgelegt von
M.Sc. Electrical and Computer Engineering
Cem Bila

an der Fakultät IV – Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
– Dr.-Ing. –

Promotionsausschuss:

Vorsitzender: Prof. Dr. Jean-Pierre Seifert (TU Berlin)
Gutachter: Prof. Dr. Dr. h. c. Sahin Albayrak (TU Berlin)
Gutachter: Prof. Dr.-Ing. Klaus Bogenberger (TU Muenchen)
Gutachter: Prof. Dr. Ing. h. c. Miroslav Svitek (Czech Technical University in Prague)

Tag der wissenschaftlichen Aussprache: 19. Mai 2022

Berlin 2022

Abstract

The road environments of the future smart cities will be more digitized and connected. Many entities and applications within this highly digitized urban environment are to communicate with one another, in order to realize intelligent decision-making processes. The fact that most of the smart city applications and devices are connected that provisions uninterrupted communication bit-pipes places Information and Communication Technologies (ICT) and network service providers in a very important position. Hence, many decision instances, which to-date did not need to take the ICT into account, now strongly depend on the availability of communication services.

In the face of recent technological developments, it becomes evident that autonomous and connected vehicles will be an integral part of this smart city ecosystem and Cooperative Intelligent Transport Systems (C-ITS). When widely adopted and operated as an on-demand mobility service, the envisioned fully autonomous and connected vehicles will bring benefits such as increased road safety, transport efficiency, and passenger comfort. Like many other smart city services, the seamless operation of automated and connected driving applications, i.e., vehicle platooning, advanced driving, extended sensors, and remote driving, will require reliable and uninterrupted connectivity. One way to realize this would be to capture the interrelation between the route planning component of autonomous and connected vehicles and the network management system of communication service providers.

The planning and control instances deployed at both ends of this interrelation would mutually benefit each other. By using the advantage of the hierarchical and deterministic route and motion planning mechanism of autonomous vehicles, the network resources can be proactively allocated along the vehicles' planned trajectories. This preemptively initiated resource allocation process based on the programmatically determined routes of autonomous and connected vehicles would definitely contribute to achieving connectivity with the required level of Quality of Service (QoS). Complementing the trajectory-based network management mechanism, the autonomous vehicles' route planning component can take into account the communication network status information. In this way, the routes of vehicles are to be optimized and adjusted by considering both traffic-related

and communication network-related conditions, which is addressed in this work. The network-aware route planning mechanism would enable vehicles to travel along roads with better network availability and higher signal strength.

In this direction, we design and implement a modular framework that allows users to model an urban scenario comprising the C-ITS elements such as autonomous and connected vehicles, wireless access points, dynamic data storage entity. By means of the established simulation environment, (near) real-time traffic and network status information can be collected from the modeled road environment. Using this dynamic data collected from the urban scenario, a group of vehicle instances representing autonomous and connected vehicles are rerouted based on the Multi-Objective Evolutionary Algorithm (MOEA) approach.

The developed modular framework enables to import and remove different objectives and constraints easily to/from the route optimization problem model. In this way, different route optimization problem instances can be defined and solved depending on the modeled scenario. The problem objectives include conventional metrics such as traveled distance, travel time, and congestion level on the roads. Additionally, communication network quality information can be incorporated into the optimization problem model as an unconventional metric. An adjustable mix of different objectives and constraints can be considered in the dynamic route planning process. A dynamic data storage entity, representing the Local Dynamic Map (LDM) component of the C-ITS, is implemented and extended with Network Context (NC) object. The provided network data layer enables the maintenance of (near) real-time network status information on the road environment. In this manner, the network conditions are taken into account by the route planning methodology, which is highly critical for the seamless operation of many automated vehicle applications such as vehicle platooning, advanced driving, extended sensors, and remote driving. A mechanism to calculate the projected congestion contribution on each road segment is implemented, which reflects the future congestion levels on the roads. Considering this additional metric as an objective or constraint would contribute to achieving global traffic optimization rather than routing the vehicles in a greedy way. The Multi-Objective Evolutionary Algorithms (MOEAs) used to reroute the vehicles, such as NSGA-2 and NSGA-3, efficiently generate a set of optimal and trade-off solutions for the vehicles.

Zusammenfassung

Die Straßenumgebungen der Smart Cities der Zukunft werden stärker digitalisiert und vernetzt. Viele Entitäten und Anwendungen innerhalb dieser hochdigitalisierten urbanen Umgebung sollen miteinander kommunizieren, um intelligente Entscheidungsprozesse zu realisieren. Die Tatsache, dass die meisten Smart-City-Anwendungen und -Geräte verbunden sind, die ununterbrochene Kommunikationsbitpipes bereitstellen, verschafft Informations- und Kommunikationstechnologien (IKT) und Netzwerkdienstanbietern eine sehr wichtige Position. Daher hängen viele Entscheidungsinstanzen, die bisher die IKT nicht berücksichtigen mussten, stark von der Verfügbarkeit von Kommunikationsdiensten ab.

Angesichts der jüngsten technologischen Entwicklungen wird deutlich, dass autonome und vernetzte Fahrzeuge ein integraler Bestandteil dieses Smart-City-Ökosystems und kooperativer intelligenter Verkehrssysteme (C-ITS) sein werden. Wenn die geplanten vollständig autonomen und vernetzten Fahrzeuge weit verbreitet und als On-Demand-Mobilitätsdienst eingeführt und betrieben werden, werden sie Vorteile wie erhöhte Verkehrssicherheit, Transporteffizienz und Fahrgastkomfort bringen. Wie viele andere Smart-City-Dienste erfordert der nahtlose Betrieb von automatisierten und vernetzten Fahranwendungen, d. h. Fahrzeug-Platooning, Advanced Driving, Extended Sensors und Remote Driving, eine zuverlässige und unterbrechungsfreie Konnektivität. Eine Möglich- keit, dies zu realisieren, besteht darin, die Wechselbeziehung zwischen der Routenplanungskomponente autonomer und vernetzter Fahrzeuge und dem Netzmanagementsystem von Kommunikationsdienstanbietern zu erfassen.

Die an beiden Enden dieses Zusammenhangs eingesetzten Planungs- und Steuerungsinstanzen würden sich gegenseitig nützen. Durch die Nutzung des Vorteils des hierarchischen und deterministischen Routen- und Bewegungsplanungsmechanismus autonomer Fahrzeuge können die Netzwerkressourcen proaktiv entlang der geplanten Trajektorien der Fahrzeuge zugewiesen werden. Dieser präventiv initiierte Ressourcenzuweisungsprozess basierend auf den programmatisch bestimmten Routen autonomer und vernetzter Fahrzeuge würde definitiv dazu beitragen, eine Konnektivität mit der erforderlichen Dienstgüte (QoS) zu erreichen. Ergänzend zu dem trajektorienbasierten Netzwerkver-

waltungsmechanismus kann die Routenplanungskomponente der autonomen Fahrzeuge die Statusinformationen des Kommunikationsnetzwerks berücksichtigen. Auf diese Weise sollen die Fahrwege von Fahrzeugen unter Berücksichtigung sowohl verkehrstechnischer als auch kommunikationsnetzbezogener Gegebenheiten optimiert und angepasst werden, was in dieser Arbeit behandelt wird. Der netzwerkfähige Routenplanungsmechanismus würde es Fahrzeugen ermöglichen, auf Straßen mit besserer Netzwerkverfügbarkeit und höherer Signalstärke zu fahren.

In dieser Richtung konzipiere und implementiere ich ein modulares Framework, das es Nutzern ermöglicht, ein urbanes Szenario bestehend aus den C-ITS-Elementen wie autonome und vernetzte Fahrzeuge, drahtlose Zugangspunkte, dynamische Datenspeicher usw. zu modellieren. Mittels der etablierten Simulationsumgebung können (fast) Echtzeit-Verkehrs- und Netzwerkstatusinformationen aus der modellierten Straßenumgebung gesammelt werden. Unter Verwendung dieser dynamischen Daten, die aus dem urbanen Szenario gesammelt wurden, wird eine Gruppe von Fahrzeuginstanzen, die autonome und vernetzte Fahrzeuge repräsentieren, basierend auf dem Ansatz des Multi-Objective Evolutionary Algorithm (MOEA) umgeleitet.

Das entwickelte modulare Framework ermöglicht das einfache Importieren und Entfernen verschiedener Zielsetzungen und Einschränkungen in das/aus dem Routenoptimierungsproblemmodell. Auf diese Weise können je nach modelliertem Szenario unterschiedliche Problemfälle der Routenoptimierung definiert und gelöst werden. Zu den Problemzielen gehören konventionelle Metriken wie zurückgelegte Distanz, Reisezeit und Staupegel auf den Straßen. Zusätzlich können Informationen zur Netzwerkqualität als unkonventionelle Metrik in das Optimierungsproblemmodell aufgenommen werden. Bei der dynamischen Routenplanung kann ein einstellbarer Mix aus unterschiedlichen Zielen und Randbedingungen berücksichtigt werden. Eine dynamische Datenspeichereinheit, die die Local Dynamic Map (LDM)-Komponente des C-ITS darstellt, wird implementiert und mit einem Network Context (NC)-Objekt erweitert. Die bereitgestellte Netzwerkdatenschicht ermöglicht die Pflege von (fast) Echtzeit-Netzwerkstatusinformationen über die Straßenumgebung. Auf diese Weise werden die Netzwerkbedingungen durch die Routenplanungsmethodik berücksichtigt, die für den reibungslosen Betrieb vieler automatisierter Fahrzeuganwendungen wie Fahrzeug-Platooning, Advanced Driving, Extended Sensors und Remote Driving sehr wichtig ist. Es wird ein Mechanismus zur Berechnung des projizierten Staubeitrags auf jedem Straßenabschnitt implementiert, der die zukünftigen Staupegel auf den Straßen widerspiegelt. Die Berücksichtigung dieser zusätzlichen Metrik als Ziel oder Einschränkung würde dazu beitragen, eine globale Verkehrsoptimierung zu erreichen, anstatt die Fahrzeuge auf gierige Weise zu routen. Die

Multi-Objective Evolutionary Algorithms (MOEAs), die verwendet werden, um die Fahrzeuge umzuleiten, wie NSGA-2 und NSGA-3, erzeugen effizient eine Reihe von optimalen und Kompromisslösungen für die Fahrzeuge.

Acknowledgments

First and foremost, I would like to express my very great appreciation to my advisor Prof. Dr. Dr. h.c. Sahin Albayrak for allowing me to conduct my Ph.D. research study at GT-ARC (German-Turkish Advanced Research Centre for ICT), an affiliated institute of Technical University of Berlin (TU Berlin). I am thankful to Prof. Albayrak that he has always provided us a perfect research environment, and supported our research and project activities. My grateful thanks are for my advisor Prof. Albayrak as he has provided patient guidance, enthusiastic encouragement and useful critiques throughout this research study. I want to thank Professor that by taking part in the research projects jointly carried out by GT-ARC, DAI-Labor (Distributed Artificial Intelligence Laboratory) and TU Berlin, I have found a chance to deepen my knowledge in the concepts of Smart City, Autonomous and Connected Driving, Internet of Things (IoT), Industry 4.0 - Smart Factory, Intermodal Mobility Assistance, Artificial Intelligence which would build and shape our future. It was an honor and opportunity to be a member of the research institute established and directed by Prof. Albayrak, and to conduct this research study under his supervision. My grateful thanks are also for Prof. Ali Abur, my Master's thesis supervisor.

It was a pleasure and an honor for me that Prof. Dr.-Ing. Klaus Bogenberger and Prof. Dr. Ing. h.c. Miroslav Svitek have joined the thesis committee. I am grateful to them as they have accepted to be a committee member and reviewed my thesis report. I would like to thank them for providing their valuable insights, comments, and advices regarding my research study.

I would like to express my special thanks and gratitude to Dr. Fikret Sivrikaya for providing his valuable and constructive insights throughout this research study. He has shared his ideas and spent his time to discuss on various research and project related issues with me. I am impressed that he has always explained the research topics we are discussing on in a very systematic order and in an understandable way. It was also an honor and opportunity to take part in the projects directed by him, to make many valuable discussions with him, and to be guided by him throughout this research study.

I would like to thank Dr. RER. OEC. Nuri Kayaoglu for providing his suggestions

throughout this research work and for providing information on the academic process in Germany. I am grateful to Dr. Kayaoglu that he has also contributed well to providing us this perfect research environment, and I want to express my thanks to him as he has always motivated and encouraged us to complete our research programs. Additionally, in general regardless of the academy, thanks to Dr. Kayaoglu that he has helped and supported us in adapting to life in Germany.

Further, I want to express my deep gratitude to Dr.-Ing. (Habil) Manzoor Ahmed Khan for providing his insights and suggestions during the planning and development of this research work. His collaboration, insights, and guidance were valuable and contributed well to completing this study.

I wish to acknowledge the help provided by Dr. Hüseyin Küsetogullari. I thank him for his valuable technical support and for sharing his insights regarding the optimization field. The discussions we have made together, the research articles and resources he has recommended were valuable and helpful for this study.

I would also like to extend my thanks to my colleagues with whom we have collaborated in the research projects. Especially, I have learned a lot from my discussions with Dr.-Ing. Marco Lützenberger, Xuan-Thuy Dang, Cem Akpolat, Christian Kuster, Dr.-Ing. Tobias Küster, Ogün Yurdakul, Martin Berger, Christian Rakow, Anon Mall and Marius Schulz. I want to thank them for their time and understanding.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation and Objectives

Autonomous driving is an emerging paradigm, which will transform the way we move and reshape our mobility ecosystem. Throughout the recent years, there has been a significant, joint effort from both industry and academy to reach different levels of autonomy for the envisioned intelligent vehicles of the future. When this revolution is fully realized, people will completely be relieved from all the driving tasks, and the intelligent vehicle platform will autonomously drive itself in every road, traffic, and weather condition. This paradigm shift is believed to bring various benefits, including improved road safety, traffic efficiency, reduced emission, and increased comfort for the passengers. Additionally, autonomous driving would have societal and economic impacts in a positive manner. It would provide an opportunity for elderly or disabled people to safely transport in their cities which would contribute to attaining an inclusive society. People would be freed from the stress of driving and be more productive during their travel time. Moreover, new employment opportunities can be created with the changing business models of the automotive sector.

One very important requirement of autonomous driving is continuous and seamless connectivity. To realize many autonomous driving applications and services, regarding road safety, traffic efficiency, or infotainment, vehicles need to have uninterrupted network connectivity with particular Quality of Service (QoS) levels. The autonomous vehicles have to communicate with other vehicles, the surrounding roadside infrastructure, and remote cloud platforms efficiently and in real-time such that these applications perform sufficiently well. For instance, the report of the 3rd Generation Partnership

Project (3GPP) group presents the connectivity requirements regarding four categories of vehicular applications: vehicle platooning, advanced driving, extended sensors, and remote driving [1]. Based on this report, the performance requirements of the advanced autonomous driving use cases, listed under these four categories, can include even $3ms$ of maximum end-to-end latency, $1000Mbps$ data rate, or 99.999% reliability. These highly stringent connectivity requirements inevitably reveal that the telecommunications industry will be a key stakeholder within the autonomous driving era. The specified QoS requirements in regard to achieving full autonomy can be guaranteed by a tight integration and close cooperation between the autonomous vehicles and network management entities of the communication service providers.

This interrelation can be captured by a possible feedback loop between the route planning component that guides the autonomous vehicles and the Software Defined Networking (SDN) controllers of the network operators, as depicted in Figure 1.1 [2]. According to this feedback loop, the interaction between the decision instances deployed at both ends will mutually affect one another and contribute to meet the connectivity requirements of envisioned autonomous driving applications. When interface (1) is considered, the planned trajectory of the autonomous vehicle is to be provided to the network operators. Based on this information, the Software Defined Networking (SDN) controllers are to proactively allocate network resources along the planned trajectories of the autonomous vehicles. By means of enabling this preemptive resource allocation, the required service quality levels would be achieved to ensure the safe operation of various autonomous driving functions. On the other hand, interface (2) allows for network connectivity information to be incorporated into the route planning mechanism. In more concrete terms, the Local Dynamic Map (LDM) of the vehicles is to be extended with a Network Context (NC) object, comprising both static and dynamic context information regarding the available access technologies [2]. With the network status information received, the route planning component is to calculate and adjust the routes of the autonomous vehicles by considering the network-related parameters, i.e., available bandwidth, latency, jitter, and packet loss.

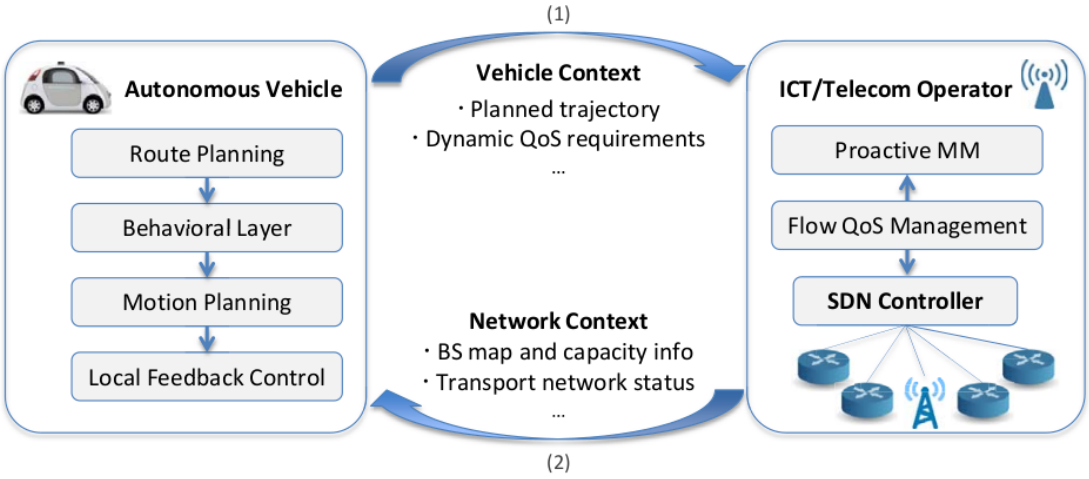

Figure 1.1: Feedback loop between autonomous vehicle route planning platform and communication network operator [2].

In this study, we mainly focus on the route planning aspect of this close cooperation between autonomous vehicle's decision-making mechanism and communication network management system. We propose a network-aware route planning policy for the envisioned autonomous vehicles to ensure uninterrupted connectivity and guarantee enhanced QoS levels along their trajectories. The conventional vehicular route planning approaches mostly rely on minimizing distance, travel time, or economic cost; however, when the unique operation principles of the connected and automated vehicles are concerned, these approaches might need to be revisited. In addition to these conventional metrics, the next-generation route planning architectures have to be context-aware and need to take other metrics into account while calculating optimum routes. When the stringent connectivity requirements of the advanced autonomous vehicle applications are examined, we believe that one of these additional metrics could be network quality information.

In light of the above considerations, the main objective of this work is to design and implement a modular framework that enables the collection of traffic and communication network status information from the road environment, and reroute the subscribed vehicles based on a multi-objective route planning mechanism. The developed and proposed

framework allows users to model both synthetic and real-world road network models, create background traffic, and instantiate a group of vehicles dynamically rerouted based on the predefined multi-objective route optimization methodology. Additionally, wireless access points such as roadside units can be deployed in the scenario which enables wireless vehicular communication and collection of communication network status information. A multi-layer data storage entity modeled and implemented, which keeps up-to-date data regarding both traffic and communication network status based on the periodically collected measurements from the road environment. Relying on this simulation framework, a group of subscribed vehicles representing automated and connected vehicles is rerouted by using the adapted Multi-Objective Evolutionary Algorithm (MOEA) approach.

1.2 Contributions

Firstly, we develop and present a modular framework, which enables us to incorporate different objectives or constraints into the route planning problem model of the automated and connected vehicles. Based on the modular design, one can easily incorporate/remove any objective and constraint to/from the optimization problem model. In addition to the conventional metrics/objectives of minimizing the total travel distance, total travel time, total traffic congestion level on the route, we take the communication network quality metric into account. By defining different groups of these objectives and constraints, we have considered different instances of the vehicular route planning problem. The developed modular framework provides us the ability to have an adjustable mix of those metrics in the objective function, depending on the scenario. The route optimization problem instances are dynamically solved for a group of vehicles in the scenario.

Secondly, a network-aware route planning policy is implemented and proposed in this work, considering the highly demanding connectivity requirements of various automated vehicle applications. In addition to conventional route planning metrics such as traveled distance, travel time, or economic cost, we incorporate the network quality information as an additional metric to our problem model. The Local Dynamic Map (LDM) data

structure is extended with Network Context (NC) object which provides up-to-date network status information regarding the wireless access technologies deployed in the modeled road environment. By this way, the optimum routes are calculated by taking into account the network quality information, i.e., signal coverage, network connectivity level, measured signal power level, along the roads, which is critical for the safe operation of automated and connected vehicles.

Thirdly, we include a metric, namely projected congestion contribution, into the route planning model that can be used as another objective to be minimized or as another constraint. After an optimum route is calculated for a vehicle upon its request, our model assigns projected congestion contribution weights along the roads of the calculated route specifically for this vehicle. The same assignment is done for every vehicle at the time that they receive a route from the remote application server, and these assigned values are continuously updated as the vehicles change to another edge. The projected congestion contribution weight quantifies the level of congestion contribution that a vehicle would produce on every road segment along its calculated route. In our model, the congestion contribution weight assigned to a road segment decreases as the estimated time that the vehicle enters into the road segment increases. In this way, the future congestion levels on each road segment are considered when calculating an optimum route for every vehicle, which leads to global traffic optimization.

Further, we have formulated the route planning of automated and connected vehicles as a multi-objective optimization problem. A population-based metaheuristic technique is adapted to solve the multi-objective route planning problem model in a dynamic road network scenario. More specifically, state-of-the-art multi-objective evolutionary algorithms, such as Non-Dominated Sorting Genetic Algorithm 2/3 (NSGA-2/3), are adapted to our problem. It is observed and verified that the adapted NSGA-2/3 algorithms efficiently generate a set of optimal and trade-off solutions for the vehicles that send requests for an alternative route.

Finally, a comprehensive simulation setup is established which integrates two state-of-the-art simulation tools for mobility and network: (i) Simulation of Urban Mobility

(SUMO) and (ii) Objective Modular Network Testbed in C++ (OMNET++). The Vehicles in Network Simulation (Veins) framework is used to integrate these simulation tools and to enable two-way, online interaction between them. This way, a realistic traffic simulation and vehicular network simulation are provided for any modeled scenario.

1.3 Thesis Organization

The rest of this thesis is organized under seven chapters as follows:

- Chapter 2 - In the first part of Chapter 2, we give background information regarding Cooperative Intelligent Transport Systems (C-ITS), summarize standardization activities to enable wireless vehicular communication by pointing out the QoS requirements of Vehicle-to-Everything (V2X) based applications, and introduce the Local Dynamic Map (LDM) concept as one of the main enablers of dynamic route planning system of future urban scenarios. The second part of Chapter 2 gives an overview of automated and connected vehicles, by focusing on their route planning mechanism.

- Chapter 3 - This chapter presents the relevant state-of-the-art approaches from the literature in regard to vehicular route planning. We provide a detailed summary of vehicular route planning techniques in the context of C-ITS. Among few research studies concerning network-aware route planning, we select and summarize two of them.

- Chapter 4 - We present the overall route planning framework and describe the main components of the proposed urban road environment model in this chapter. Additionally, the vehicular route planning problem is mathematically modeled as a multi-objective optimization problem that is to be dynamically solved via a population-based metaheuristic technique.

- Chapter 5 - We introduce and present our evolutionary computation approach to solving the multi-objective vehicular route planning problem with constraints. At

the beginning of this chapter, we present our results obtained after applying a genetic algorithm to a single-objective route planning problem. In this preliminary section, we examine how the genetic algorithm works for the single-objective optimization problem instance with different algorithm-specific configurations, i.e., population size, crossover type, mutation rate, selection type. The following sections present the multi-objective evolutionary algorithm approach. Further, it details one of the state-of-the-art methods in this context, namely NSGA-2, which is adapted and applied to find optimum route solutions for the vehicles in a dynamic road network scenario.

- Chapter 6 - This chapter is reserved for the implementation details of our proposed modular framework, which enables us to import and remove different optimization objectives or constraints for the dynamic vehicular route planning problem model. First, we describe the integrated and bidirectionally coupled simulation framework used in this study. The integrated simulation environment is enabled by coupling the well-known mobility simulator, Simulation of Urban Mobility (SUMO), and widely used network simulator, Objective Modular Network Testbed in C++ (OMNET++), by means of the Vehices in Network Simulation (Veins) framework. After describing how this integrated simulation framework functions, we explain how it is extended to enable dynamic, and multi-objective vehicular route planning by considering both traffic and network-related metrics.

- Chapter 7 - The simulation test results, obtained through a set of simulations carried out with different configurations, are presented. The abstract road network models provided by the SUMO mobility simulator and a real-world model are used for the simulation tests. It is aimed to observe if the applied multi-objective route planning methodology can periodically provide optimum route solutions for the vehicles, by considering dynamically changing traffic and communication network metrics. Based on the simulation results, it is concluded that the adapted multi-objective evolutionary algorithm efficiently finds out a set of optimum route

solutions for the vehicles. The subscribed vehicles, vehicles that are periodically requesting a new route alternative, are rerouted based on the calculated route by the multi-objective route optimization algorithm. The optimal routes are calculated by taking into account the metrics such as distance, traffic congestion, and communication network quality. The resulting simulation outcomes are presented and evaluated for different problem instances and configurations.

- Chapter 8 - The final chapter concludes the thesis by summarizing what we have achieved as a result of this study and proposes a few research directions for future work.

2 Background Information

This chapter gives background information on Cooperative Intelligent Transport Systems (C-ITS) and the concept of Local Dynamic Map (LDM). Additionally, we overview the hierarchical path planning mechanism of automated and connected vehicles. Further, the operation of autonomous and shared mobility services is briefly introduced in the context of the dynamic vehicular route planning problem.

2.1 Cooperative Intelligent Transport Systems (C-ITS)

This section introduces the Cooperative Intelligent Transport Systems (C-ITS) as the main enabler of various automated and connected driving use cases, based on wireless inter-vehicular communication. C-ITS refers to a system in which different constituents of urban mobility, i.e., vehicles, roadside infrastructures, central traffic management entities, personal devices, communicate and cooperate to improve road safety, traffic efficiency, and driving comfort. In this digitized and cooperative framework, the exchange of road data among traffic participants paves the way to realizing a wide variety of ITS services in higher standards.

To exploit the benefits of C-ITS, it is a key requirement to standardize communication architecture and provide interoperability among different digitized components on the road environment. The standardization bodies European Telecommunications Standards Institute (ETSI) [3], European Committee for Standardization (CEN) [4], Internet Engineering Task Force (IETF) [5], International Organization for Standardization (ISO) [6] have contributed to standardize different communication aspects of the C-ITS framework in close collaboration with the CAR 2 CAR Communication Consor-

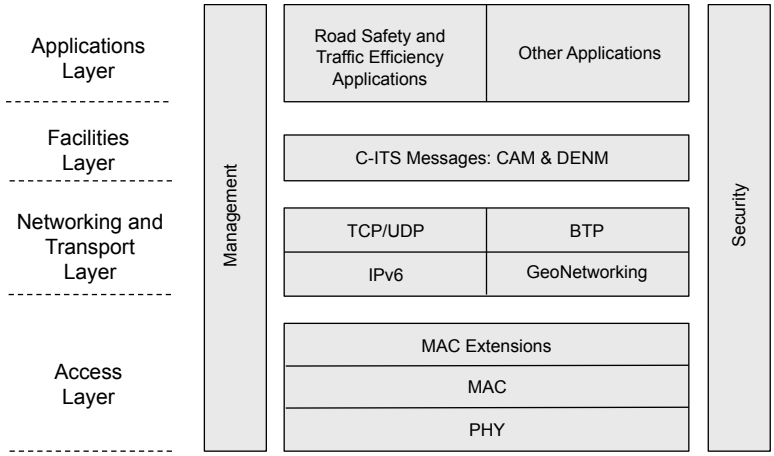

Figure 2.1: C-ITS Protocol Stack [9].

tium (C2C-CC) [7], as detailed in the articles [8, 9]. The results obtained from major European projects employing C-ITS architectures, i.e., SAFESPOT [10], Co-operative Vehicle-Infrastructure Systems (CVIS) [11] and DRIVE C2X [12], have also contributed to this standardization. In the following we detail the C-ITS standards determined by the ETSI organization. This section focuses on the ITS-G5 station specified by the ETSI group, and for its U.S counterpart of Wireless Access in Vehicular Environments (WAVE) standardized by IEEE, readers can be referred to [13]. Additionally, in the next sub-section we will outline the standardization of cellular V2X (C-V2X) communication, mainly carried out by the 3rd Generation Partnership Project (3GPP) group.

2.1.1 ETSI C-ITS Standardization

C-ITS communication standards are presented in ETSI EN 302 665 [14] and ISO 21217 [15]. The ITS station reference architecture is described in [14]. This commonly accepted communication architecture can be deployed on different types of C-ITS sub-systems: vehicle ITS sub-system, roadside ITS sub-system, personal ITS sub-system, and central ITS sub-system. All of these four ITS sub-systems are to contain an ITS station and the functional components of these stations are detailed in the ETSI report [14].

ETSI C-ITS Communication Architecture / Protocol Stack

The C-ITS protocol stack for vehicle and roadside ITS stations is depicted in Figure 2.1, with four C-ITS layers: access layer, networking and transport layer, facilities layer and application layer [9]. The access layer of the C-ITS protocol stack comprises the physical layer and data link layer of the Open Systems Interconnection model (OSI model). There are different communication interfaces available in the access layer, i.e., ITS-G5, WiFi, GPS, BlueTooth, 2G/3G/.. and Ethernet [14]. We will focus on the ITS-G5 communication technology, which is based on the IEEE 802.11p and forms a basis for various vehicular applications. As indicated in the ETSI EN 302 571 Release, three ITS frequency bands are reserved at 5GHz band for ITS-G5 (see Figure 2.2): ITS-G5B (5.855 GHz to 5.875 GHz), ITS-G5A (5.875 GHz to 5.905 GHz) and ITS-G5D (5.905 GHz to 5.925 GHz) [16]. The ITS-G5B band spans 20MHz with two service channels of 10 MHz dedicated to non-safety C-ITS applications [9]. The ITS-G5A spans a 30 MHz band with one control channel and two service channels of 10 MHz [9]. The control channel (5.895 GHz to 5. 905 GHz) is reserved as the primary safety channel [9]. The ITS-G5D band, spanning 20 MHz with two service channels, is reserved for future C-ITS systems [9]. As specified in the related ETSI report [16], the maximum power spectral density is to be 23 dBm/MHz. The dedicated access technology for the time-critical, safety-related applications is based on IEEE 802.11p for the access layer of the C-ITS architecture [8] [17]. ITS-G5 stations use a basic ad hoc mode, namely Outside the Context of a BSS (OCB), where BSS refers to Basic Service Set in the IEEE 802.11 wireless networks [9]. The OCB ad hoc mode avoids the management procedures, i.e., channel scanning, authentication, association, and allows for a direct and immediate message transmission without time-consuming delays [9]. A Decentralized Congestion Control (DCC) mechanism is used in the 5GHz range to avoid channel congestion during the vehicular ad hoc communication [16] [18] [19]. The carrier sense multiple access with collision avoidance (CSMA/CA) scheme is used by the C-ITS stations as in the IEEE 802.11p standard [20] [21].

The networking and transport layer of the C-ITS communication architecture, rep-

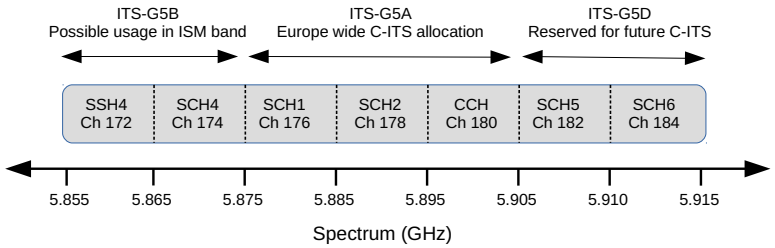

Figure 2.2: C-ITS Spectrum [9] [17].

resenting the OSI model's third and fourth layers, is responsible for transferring data between source and destination ITS stations by utilizing the capabilities provided by the underlying access layer [8]. The GeoNetworking protocol, executed in the GeoAdhoc router, enables the delivery of packets in an ad hoc network based on geographical addressing, which supports both point-to-point and point-to-multipoint transmission [22]. By means of the addressing capabilities, it is possible to send a packet to an individual C-ITS station by specifying its geographical position (GeoUnicast) or distribute a message inside a geographical area of a specific geographical shape, i.e., circular area, rectangular area, elliptical area (GeoBroadcast/GeoAnycast) [9] [23]. Additionally, it is possible to transmit packets to all nodes in one-hop or n-hop neighborhoods without using geographical addressing, referred to as single-hop broadcast and topologically-scoped broadcast [9] [24]. There are three forwarding algorithms specified in the ETSI EN 302 636-4-1 document [22]: i) simple GeoBroadcast forwarding algorithm, ii) area contention-based forwarding algorithm, and iii) area advanced forwarding algorithm. The Basic Transport Protocol (BTP), residing between the GeoNetworking protocol and facilities layer, provides an end-to-end and connection-less communication service in the vehicular ad hoc networks [25]. It also takes a role in multiplexing and demultiplexing the messages at the ITS facilities layer [25]. In addition to the GeNetworking protocol, Internet Protocol Version 6 (IPv6) is also employed by the C-ITS stations over cellular networks depending on the application [26].

The facilities layer, located between the networking and transport layer and applications layer, stands for the fifth, sixth, and seventh layers of the OSI model [14]. It

supports the application development by providing standardized access to data, information and common functionalities [8]. In the facilities layer, the two C-ITS message types, cooperative awareness message (CAM) and decentralized environmental notification message (DENM), are standardized in ETSI EN 302 637-2 [27] and ETSI EN 302 637-3 [28] respectively [29]. The CAM messages are periodically sent to the neighboring nodes, which include status information such as position, vehicle speed, destination, and acceleration. As indicated in [28], all the ITS stations are enabled to generate, send or receive CAM messages when they participate in the IEEE 802.11p ad hoc network. The C-ITS stations receiving the CAM messages are to be aware of the positions, movement, and sensor information of the neighboring nodes [28]. The DENM messages are sent in the case that a specific safety-critical event or a specific traffic event is detected in the driving environment. The C-ITS stations receiving the DENM messages are notified about the event and the necessary actions are taken to avoid any possible hazardous situation. As described in [28], upon detection of any hazardous traffic event, the corresponding C-ITS station detecting the event starts broadcasting DENM messages to the C-ITS stations inside the relevant geographical area. The transmission of DENM messages continues with a certain frequency until the detected event does not exist anymore [28]. The broadcast of DENM messages is terminated automatically either when the event disappears after a predefined expiry time passes, or the broadcasting C-ITS station sends a specific DENM message indicating that the event disappeared [28]. The message formats for CAM and DENM are detailed in ETSI reports EN 302 637-2 v1.3.1 [30] and EN 302 637-3 v1.2.1 [31] respectively based on the common data dictionary presented in ETSI TS 102 894-2 v1.3.1 [32]. Finally, it is worth noting that Local Dynamic Map (LDM) is a core functional component of the facilities layer in regard to information support. LDM allows to fuse data from different sources, keep the collected information up to date and build the local data model of the local road environment [33], which will be further detailed in the next section.

The highest layer of the C-ITS protocol stack, namely applications layer, mainly defines three classes of ITS applications: road safety, traffic efficiency, and other appli-

cations, which are introduced as Basic Set of Applications (BSA) in the ETSI TR 102 638 v1.1.1 report [33]. The catalog of ITS applications and corresponding use cases are listed in the report for each of the three categories. In the ETSI report, when the requirements of the V2X based applications are examined, the minimum frequency of the periodic messages is mostly stated as $10Hz$, and the maximum tolerable latency is given as $100ms$. The security layer is in charge of avoiding possible attacks on the C-ITS systems such as 'extraction/modification of secret material, tampering with the vehicle's ITS station, network jamming, alteration attack, fake message injection, Sybil attack and privacy attack [8]'. As indicated in the ETSI EN 302 665 report, the functional blocks of the security layer include firewall and intrusion management, authentication, authorization and profile management, identity, crypto key and certificate management, common security information base (SIB), and hardware security modules (HSM) [14]. Lastly, the management layer of the C-ITS reference architecture allies applications, manages networks, security functions, and dynamic interface selection, which is described in [8]. The main tasks of the management layer are grouped into four functional blocks as detailed in the ETSI EN 302 665 document [14]: regulatory management, cross-layer management, station management, and application management.

2.1.2 3GPP Cellular Vehicle-to-Everything (C-V2X) Communication Standardization

In this section, we summarize the cellular-based vehicular communications standards and the supported applications. The cellular V2X communication technology is the other key enabler for vehicular applications together with the Dedicated Short Range Communication (DSRC) and ITS-G5. The standardization activities regarding C-V2X communication is mainly lead by the 3rd Generation Partnership Project (3GPP) organization that unites various standardization organizations (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC) to produce specifications and technical reports regarding cellular communication systems, including Cellular Vehicle-to-Everything (C-V2X) communication [34]. The global and cross-industry organization, 5G Automotive Association (5GAA) [35],

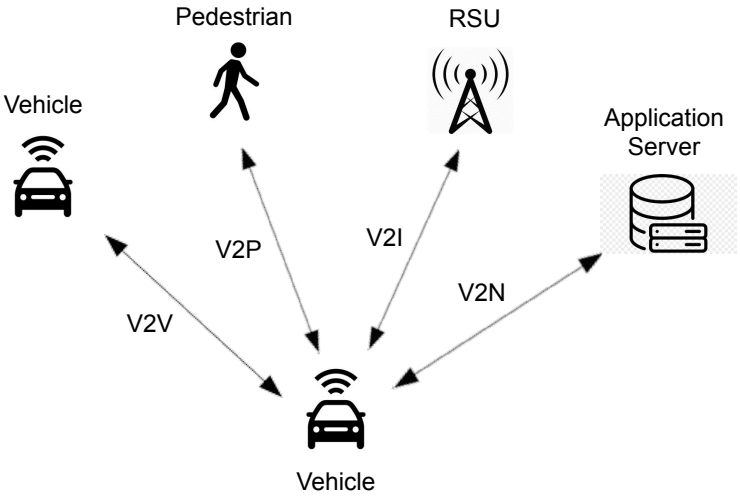

Figure 2.3: 3GPP C-V2X standards support V2V, V2P, V2I and V2N applications [36].

supports the standardization efforts of the 3GPP within this context. 5GAA incorporates various companies from automotive (AUDI AG, BMW AG, BMW Group, Daimler AG) and telecommunications (Ericsson, Huawei, Intel, Nokia, Qualcomm) industries to develop solutions for future connected mobility [35].

3GPP V2X Application Support: V2V, V2P, V2I and V2N

3GPP defines four types of communication support for the V2X applications: Vehicle-to-Vehicle (V2V), Vehicle-to-Pedestrian (V2P), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Network (V2N) [36] [37] [38] (see Figure 2.3). As indicated in the 3GPP TS 22.185 (Release 15) [36], these entities (vehicles, roadside infrastructures, pedestrians, application server) can employ co-operative awareness to make more intelligent services possible, i.e., autonomous driving, vehicle warning, traffic management [37]. The V2V-based applications support UEs in proximity to exchange messages including V2V application information (e.g., location, vehicle attributes, and traffic information) [36]. 3GPP standard 3GPP TS 22.185 indicates that UEs involving in the V2V based applications need to have subscriptions and authorization from the network operator. The message transmission in the V2V based applications is mostly in the form of broadcasting [36] [37]. This

message transmission in the 3GPP V2V applications can be either directly between UEs; or in the case that UEs are not within their communication range, the messages can be transmitted between UEs through infrastructures, i.e., RSUs, application servers [36]. In V2I communication-based applications, UEs supporting V2I applications transmit messages (including V2I application information) to RSUs or local application servers [36]. Then, the RSU or the local application server retransmits the received V2I application information to one or multiple UEs supporting V2I applications [36]. In this scenario, there could be one locally relevant application server that serves a particular geographical area, or multiple application servers functioning for overlapping areas, concerning the same or different V2I applications [36] [37]. Similar to V2V applications, messages (carrying V2P application information) are exchanged between vehicle UEs and pedestrian UEs in the V2P applications [36] [37]. In this case, the UE supporting V2X application in a vehicle can transmit a message to pedestrian UE to warn the vulnerable road user; or a pedestrian UE supporting V2X application can send a message to warn vehicle [36] [37]. As in the case of V2V application, the message transmission in V2P application can be directly between vehicle UE and pedestrian UE or via infrastructure, RSU, application server, etc [36]. The UEs involved in the 3GPP-based V2P applications are to be subscribed to and have authorization from a network operator. As noted in the 3GPP TS 22.185 report [36], the main difference between V2V and V2P applications is that UEs supporting V2P applications usually have a lower battery capacity. Due to this reason, the UEs supporting V2P applications might not be able to transmit messages in the periodicity compared to UEs involved in the V2V applications [36]. In the case of V2N applications, UEs supporting V2N applications communicate with the application servers via Evolved Packet Switches (EPS) [36] [37].

C-V2X Communication Interfaces: LTE-Uu and PC5 Air Interface

3GPP defines two types of communications in the vehicular domain, namely LTE-Uu and PC5. The cellular interface named LTE-Uu supports vehicle-to-infrastructure (V2I) communication, while the PC5 interface supports vehicle-to-vehicle (V2V) communica-

tions via direct LTE sidelink [39]. In the case of V2I communications by LTE-Uu, a user equipment UE (vehicle UE) transmits its message to the eNodeB via uplink; and then the same or another eNodeB is to transmit the received packet to UE far away by using unicast downlink communication or enhanced Multimedia Broadcast Multicast Service (eMBMS) [40]. The communication interface LTE-Uu provides data dissemination to a wide range by using the cellular core network [40]. A semi-persistent resource management mechanism can be used for LTE-Uu, which is to be beneficial for V2X applications by reducing the scheduling overhead [40]. The PC5 interface allows direct communication between UEs without a requirement that every packet is to pass through eNodeB [40]. There are two communication modes defined for the PC5 interface to enable advanced vehicular applications with low latency and high-reliability requirements, namely mode 3 and mode 4 [39]. The PC5 mode 3 is available when the vehicles are in the cellular coverage and the network resources are managed by the eNodeB [39]. In the case of PC5 mode 4, vehicles reserve their resources autonomously and independently from the cellular infrastructure [39] [40].

Enhancement of 3GPP Support for V2X Services (eV2X)

3GPP identifies four categories of enhanced V2X (eV2X) applications and use cases: (i) Vehicle Platooning, (ii) Advanced Driving, (iii) Extended Sensors, and (iv) Remote Driving [1] [37] [41]. These four types of enhanced V2X applications determined by the 3GPP standardization organization are overviewed in the 3GPP TR 22.886 (Release 15) [41] and 3GPP TS 22.186 [1] reports, icluding their communication requirements.

In the platooning application, vehicles dynamically form a group and travel together in a closely linked manner [1] [41]. The vehicles forming the platoon periodically receive status information (e,g speed, heading, intentions such as braking, acceleration) from the leading vehicle via wireless short-range V2V communication to maintain the feasible distance between them [1] [41]. When the inter-vehicle distance is maintained at a feasible minimum, the platooning application has the benefit of improving traffic efficiency, lowering fuel consumption, and reducing the number of drivers [41] [42]. The advanced

driving application provides semi-automated or fully-automated driving, where a longer inter-vehicle distance is presumed [41]. Vehicles and RSUs share their information collected by their local sensors or nearby vehicles with other vehicles in proximity, which allows a coordinated and synchronized driving [41]. The advanced driving application enables improved road safety, traffic efficiency and collision avoidance [37] [41]. In the extended sensors application, vehicles, roadside units, pedestrian UEs and application servers exchange raw or processed data among themselves [41]. The extended sensors application augments vehicles' perception of the environment beyond the perception obtained only by local sensors and enables more intelligent decision-making for vehicles [37] [41]. Lastly, in the case of remote driving, a remote driver or a V2X application takes control of a remote vehicle and remotely operates it [41]. The remote driving application is useful for the passengers who are not able to drive by themselves, or for saving vehicles located in dangerous situations [41]. The communication requirements of these four types of enhanced V2X based applications are detailed in the 3GPP TS 22.186 (Release 15) technical specifications report [1]. The required payload, transmit rate ($message/sec$), maximum end-to-end latency (ms), reliability (%), data rate ($Mbps$), and the minimum communication range (m) are specified for different communication scenarios for each category of application.

2.1.3 Local Dynamic Map (LDM)

The Local Dynamic Map (LDM) concept plays a central role in C-ITS and it forms a basis for autonomous driving applications. Needless to say, the highly automated and connected vehicles of the future have to precisely aware of their position and the environment around them such that they can generate the optimum trajectory plan in every traffic situation. In this sense, LDM is a key technology and enabler that integrates 3D map data, sensory data, and other traffic information received via V2X communication. By this way, LDM provides a (near) real-time, highly accurate, and dynamic representation of the road environment. The self-driving, intelligent vehicles of the future are to rely on this enhanced data integration concept to drive themselves safely

and efficiently. As highlighted in [43] LDMs provide common and consistent situational awareness for all the applications running as part of a cooperative ITS system, which eventually brings two advantages: i) consistency due to the shared common information model and ii) improved computational performance as the acquisition and fusion of source data is performed only once.

As indicated in [2], LDM concept is standardized by both ETSI [44] and ISO [45] [46] standardization organizations. Various static and dynamic traffic information, collected by Cooperative Awareness Messages (CAMs) or Decentralized Environmental Notification Messages (DENMs), are integrated into LDM structure [2] [47]. LDMs have a layered and conceptual data structure. In each layer, different types of C-ITS data and traffic information are maintained [48] [49] [50]. The first layer of LDM keeps permanent static data, such as map data, road data, and intersections. The sources of this static data are geographic information systems (GIS) and map providers [50]. The second layer of LDM contains transient static data, i.e., road infrastructures, ITS stations, and traffic signs. The third layer of LDM stands for transient dynamic data, including traffic congestion level, weather condition, and traffic light signal phase. The fourth layer of LDM keeps highly dynamic data regarding vehicles, pedestrians, etc.

2.2 Automated and Connected Vehicles

2.2.1 Hierarchical Decision Making Mechanism

The path planning of autonomous vehicles can be modeled as a hierarchical decision-making mechanism. Each layer of this decision-making mechanism handles a different level of planning task with different constraints. Inspired by [51], we categorize this decision-making mechanism of the autonomous vehicles mainly under four levels of hierarchy as depicted in Figure 2.4. From higher to lower levels, the decision-making components of the autonomous vehicles might be named route planning, behavior planning, motion planning, and local feedback control. It can be stated here that the outputs of higher-level planning components are given as input for the lower-level planning func-

tions.

At the highest level of this decision-making mechanism, global route planning (or mission planning) is utilized on a road network to calculate an optimal, high-level route from the start location to a specified destination location. Based on the calculated route in the route planning layer and by considering the rules of the road, the behavioral layer determines the local and shorter-term driving tasks, i.e., lane change, takeover, stop at the traffic lights, and turn right or left. The motion planning handles finding an optimal path or trajectory to perform the determined driving behavior by considering the safety and feasibility of the trajectory, estimated locations of the static and dynamic objects around the vehicle, and passenger comfort. Finally, the lowest feedback control layer sends appropriate commands to the actuators for the proper execution of the planned trajectory and tracks the errors during the trajectory is executed by the vehicle. In this study, we focus on the higher-level route planning part of this mechanism.

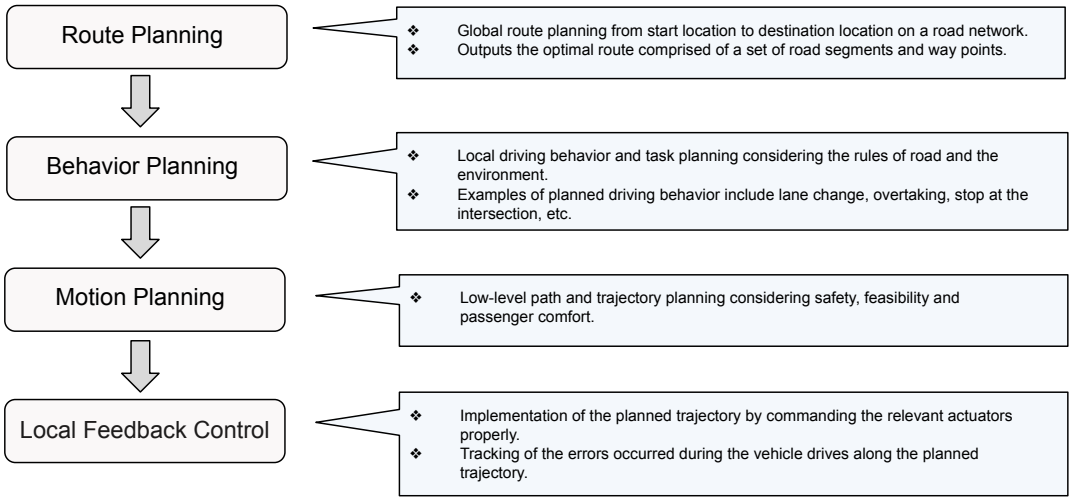

Figure 2.4: Hierarchical decision-making and planning mechanism of autonomous vehicles.

2.2.2 Autonomous Mobility on Demand (AMoD)

The global trend towards highly connected, digitized, and information-everywhere world paves a way to transformation in almost every era of human life. Being not an exception,

mobility is one of the domains that has been significantly influenced in this concern. Especially, people living in densely populated urban areas are looking for integrated, multi-modal, and flexible mobility solutions rather than traditional ways of transport, such as mass public transport or private car ownership. The behavior change in this regard can be affiliated to the recently emerging and widely accepted Mobility-as-a-Service (MaaS) concept.

The term MaaS refers to an on-demand, user-oriented and flexible mobility management system that is able to integrate a wide range of transport modes in the form of a door-to-door trip plan. Operated on a digital platform, the MaaS system suggests a solution, which is tailored to commuters' transport preferences. Based on the transport needs of users, the platform plans a chain of heterogeneous mobility services, including car-sharing, ride-hailing, public transport, taxi, etc., and provides a unified solution [52]. Additionally, the MaaS model eases other transport-related services, such as payment and subscription, for the commuters [53].

Considering the recent technological advancements towards realizing fully autonomous driving, autonomous vehicle fleets can be utilized as shared and on-demand mobility services in our cities, which is referred to as the Autonomous Mobility on Demand (AMoD) system. As indicated in [53], there are already attempts to adopt the autonomous driving technology on a wider scale as a shared, on-demand mobility service provider, i.e., nuTonomy start-up company tests its driverless taxis in Singapore [54], Uber, and Lyft have plans and ongoing tests in the same direction [55]. When AMoD concept is widely adopted, it can be presumed that there would be different operators utilizing their group of autonomous vehicles. At this point, the fleet level control and management of autonomous vehicles constitute another research dimension in this domain. In the following, we briefly introduce the system model of the autonomous and shared mobility system, describe its operation principles and identify a set of objectives that need to be considered in the context of autonomous vehicles' dynamic routing problem.

Formally speaking, an autonomous vehicle fleet to serve the travel demand of the passengers can be denoted as $F = \{v_i, v_2.., v_M\}$, where F represents the fleet of vehicles

and each vehicle v_i has a maximum passenger capacity of c_v. The state of a vehicle S_v can be represented as a tuple $S_v = \{p_v, t_v, N^p, St^v\}$, where p_v is the current position, t_v is time, N^p is the number of passengers the vehicle is serving and St^v stands for the status of the vehicle, i.e., serving, idle and mobile, parked. The requests R dynamically sent by the passengers to the fleet operator can be represented as a tuple $r_i = \{o_r, d_r, t_r\}$, where o_r is the origin node, d_r is the destination node, t_r is the time instant that request is sent by the passenger [56]. The maximum waiting time of a passenger after requesting a ride is denoted as W_t^{max}. Similar to the model presented in [56], the latest pick-up time of a passenger can be defined as $Tp^{max} = t_r + W_t^{max}$ and the earliest possible time a passenger can reach the destination can be defined as $Td^* = t_r + f_{tt}(o_r, d_r)$, where f_{tt} is a function that finds the travel time from an origin to destination node.

The control and management mechanism of an AMoD fleet has to handle thousands of requests, originating from different regions of the city. The requests have to be assigned to the available vehicles of the mobility service provider, and the vehicles with assigned routes have to be guided to their destinations efficiently. The routing algorithm of the vehicles has to consider multiple criteria, such as traveled distance, travel time, environmental impacts, and safety. In this work, we incorporate the network quality information into the route planning scheme, which would be critical for various applications utilized by autonomous and connected vehicles. While assigning the trip requests to vehicles, the predicted future demand can also be incorporated into the decision-making mechanism as highlighted in [56]. As another important point in addition to request-to-trip assignment and dynamic routing of on-duty vehicles, the idle vehicles have to be re-routed to the regions of the city where higher mobility demand is expected in the near future, which is referred to as rebalancing [57] in the literature.

3 State-of-the-Art: Literature Review on Vehicular Route Planning Approaches

The following sections present a set of recent and prominent approaches for the vehicular route planning problem. We classify the relevant works in the literature under two categories: (i) V2X-based Vehicular Route Planning Approaches in the Context of C-ITS and (ii) Network-aware Route Planning Approaches.

3.1 V2X-based Vehicular Route Planning Approaches in the Context of C-ITS

In [58] authors present and evaluate three proactive re-routing strategies, namely Dynamic Shortest Path (DSP), Random k Shortest Paths (RkSP), and Entropy Balanced k Shortest Paths (EbkSP), to alleviate the effects of congestion. As detailed in the article, the DSP algorithm is a classical route planning algorithm that considers the lowest travel time. RkSP assigns each vehicle to one of the randomly computed k paths. EbkSP, which authors propose, performs more intelligent path selection by considering the effects of the path selection on the future traffic density. The system model collects data from vehicles and also from the road-side sensors when possible. Based on the collected data, the congestion level in the directed and weighted road network is periodically evaluated. In the case that a sign of congestion is observed on certain road segments, the system provides individually-tailored re-routing guidance to vehicles that are likely to pass through the congestion. As a result of extensive simulation studies carried out by SUMO [59] and TraCI [60], authors point out that although all strategies significantly

improve the travel time, the proposed EbkSP strategy provides the most promising results. It is pointed out by the authors that EbkSP balances best the trade-offs between low average travel time and low overhead along with several parameters.

In [61] authors present a predictive traffic management scheme based on an IEEE 802.11p based V2I communication. The proposed model implements a two-layer control system. In the first layer, vehicles equipped by OBUs send data packets to RSUs, comprising information such as position, speed, and intended destination. In the second control layer, a central control unit aggregates data from the RSUs using WAN infrastructure, predicts the future congestion states by using a linear prediction model, and adaptively re-routes the vehicles to avoid congestion. Based on the simulation results, the authors indicate that their prediction-based model provides improvements in the road traffic conditions by reducing the total journey time and waiting time of the vehicles.

Likewise, authors in [62] propose an integrated route guidance model, based on short-term traffic flow prediction. In the proposed framework, the infrastructure (RSUs) acquires the position of all vehicles in their communication range, generates traffic flow information based on this collected data, and sends the obtained records to the traffic information center via the wireless communication system. The traffic information center updates the traffic database and resends the updated traffic data to the RSUs on regular basis. Vehicles then acquire this traffic database from the RSUs onto their own on-board computers. Based on this data, each vehicle unit forecasts short-term traffic flow by using Kalman Filter. Thereafter, based on the prediction results, a hierarchical and regional Dijkstra algorithm is adopted to periodically re-plan the current optimal route for the vehicles.

Authors in [63] address the congestion problem in large cities and propose a re-routing mechanism based on vehicle-to-roadside unit communication. In the proposed scheme vehicles periodically send a message, including their position, speed, route, and destination, to the nearest RSU by means of a single hop long-range communication process, such as LTE or 4G. Based on this collected data, the RSU computes the average speed

and the total number of vehicles at each edge within its coverage. Then, the RSU updates the edges' weights that are modeled as inversely proportional to the traffic condition in the road. Authors make use of the k-Nearest Neighbor algorithm (KNN) to classify congestion levels on roads; defined as slight, moderate, and severe congestion. For each congested road, an area of interest (AoI) is defined. Thereafter, if the RSU detects a vehicle is inside the AoI of the congested road, it calculates k alternative roads for this vehicle to avoid the congested road. Based on the simulation results, it is claimed by the authors that their algorithm re-routes vehicles appropriately by performing load balance among the alternative paths.

A self-adaptive interactive navigation tool (SAINT) is proposed in [64] that relies on the interaction between vehicles and vehicular cloud. In the proposed system, vehicles with the navigation system communicate with RSU or eNodeB to access the cloud, namely Traffic Control Center (TCC). Vehicles periodically report their trajectories and navigation experiences to the cloud. By using the reported vehicle trajectories, TCC estimates the near-future congestion level of each road segment. In the case that a new vehicle requests its navigation path, the TCC provides a globally optimal navigation path that bypasses highly congested, bottleneck road segments. Authors claim that SAINT outperforms the legacy navigation scheme based on Dijkstra's algorithm by using real-time road traffic snapshots. Based on simulations carried out on a road network from Manhattan, NY, it is noted that SAINT can reduce the travel time during rush hours by 19%.

In this regard, authors in [65] propose a gossiping method for information propagation in a wireless vehicular network environment to alleviate the congestion. In the proposed model, mobile vehicle agents in close proximity (agents passing one another or agents located at the same edge) communicate with each other via V2V communication and exchange traffic information regarding the road condition. The gossip network model, based on a real road map, is modeled as a non-cooperative game, meaning that each vehicle agent aims at minimizing its own journey length without trying to optimize overall network performance. Based on the simulation runs performed on a hybrid

microsimulation tool, it is concluded that the gossiping model is efficient even when a relatively low percentage of the agents are able to gossip. However, it is observed by the simulation results that when gossiping agent percentage exceeds a certain limit, the average journey length starts decreasing. Authors explain this as a ping-pong effect such that when the percentage of gossiping vehicle agents increases, most of the gossip agents change their way to the same more available roads, which eventually blocks these roads. Furthermore, it is claimed that the proposed gossiping model is found to be more efficient compared to two (offline and online) centralized information dissemination models.

Likewise, authors in [66] state the beneficial sides of using an infrastructure-free V2X communication among vehicles to avoid congestion on road networks when compared to the conventional systems mentioned in the article. In the proposed model, the V2X-enabled vehicles transmit the average speed of every passed edge to the neighboring vehicles in their vicinity. The vehicles receiving the speed information recalculates the edge weights (edge trip time) for the corresponding road segments. Subsequently, the updated edge weights are used by each vehicle to recalculate the most optimal travel route based on a conventional shortest path algorithm. In the case, that newly calculated routes are found to be shorter in travel time, the new rote is followed by the vehicle. As a result of simulations performed with the incremented V2X penetration rate (percentage of V2X-equipped vehicles in the scenario), it is found out by the authors that the proposed decentralized V2X communication method provides a considerable travel time reduction for both regular and V2X-enabled vehicles.

In the same direction, authors in [67] propose a dynamic route guidance system relying on V2V communication. The proposed system comprises two main parts, namely guidance and detour. The guidance part enables vehicles to gather real-time traffic information by means of V2V links and multi-hop relaying to find out potential better routes that will take less travel time than the shortest path. Authors make use of a detour algorithm in order to bypass void areas containing empty roads during the guiding process.

Inspired by ant behavior, authors in [68] propose an anticipatory vehicle routing strat-

egy using delegate MAS. In this decentralized approach, the proposed MAS model consists of three main entities: 1) vehicle agent; 2) infrastructure agent; and 3) virtual environment. The vehicle and infrastructure agents are responsible for coordinating traffic whereas the virtual environment corresponds to software representation of the environment. Rather than directly communicating, the vehicle and infrastructure agents send out lightweight agents, referred to as ants, that somewhat mimic the ants' behavior. Vehicle agents employ two types of ants, namely exploration ants, and intention ants. The former explores the feasible routes in the virtual environment of a city and returns estimated route durations. When a vehicle agent selects one of the explored routes, it makes this information available for other vehicles by means of the intention ants. The intention ants also provide this information to infrastructure agents such that infrastructure agents predict the future traffic loads and provide the estimated future traffic state back to vehicle agents. As a result of the simulations, authors conclude that their proposed anticipatory routing strategy, using forecast data, allows drivers to avoid congestion and prevent them from forming congestion. Additionally, it is noted that the proposed model helps drivers reach their destination 35% faster.

Authors in [69] propose a dynamic route guidance scheme to mitigate road congestion by utilizing only V2V communication. In the proposed system model, vehicles on the same road segment are organized as a cluster. A temporarily selected cluster-head vehicle collects real-time traffic information (speed, fuel consumption, vehicle density, incidents, etc.) from the cluster members, generates a message for the real-time traffic information in its segment, and shares this information with the cluster-head vehicles of other road segments. The cluster-head vehicle, receiving the message, broadcasts the message to the vehicles in its cluster. When a vehicle receives the message, the proposed scheme adopts the trust probability (TP) to determine whether the current guided route of the vehicle remains to be optimal one or not, by considering three metrics: travel time, vehicle density, and fuel consumption. Relying on the trust probability, it dynamically determines the optimal route during the travel and assists the vehicle by considering drivers' individual preferences. As a result of simulation analysis, authors claim that

their infrastructure-free route guidance scheme achieves better traffic efficiency in terms of time efficiency, balance efficiency, and fuel efficiency, in comparison with existing schemes.

Authors in [70] design and evaluate a decentralized system where vehicles crowd-sourced traffic information in an ad hoc manner and are dynamically rerouted to minimize the trip times. The proposed system architecture consists of three main modules: i) traffic sensing module ii) dissemination module iii) traffic estimation and dynamic routing module. In the traffic sensing module, traffic sample data is generated by vehicles when they exit a road segment. The dissemination module makes use of a utility function to prioritize and broadcast a subset of information by considering the bandwidth restrictions. Then, in the third module, the current traffic conditions are estimated and vehicles are dynamically rerouted by means of a modified Dijkstra algorithm. To set up the evaluation platform, authors integrate QualNet as a communications network simulator and MobiDense as a mobility simulator. Based on the simulation studies carried out on a road network in downtown Portland, OR, it is concluded that the proposed decentralized rerouting scheme can significantly reduce traffic congestion in a realistic scenario.

Authors in [71] propose a real-time global path planning algorithm based on a hybrid-ITS system that makes use of both VANET and cellular communication. Unlike individual path planning approaches, authors consider the problem from a global perspective as well as meeting drivers' preferences. In the proposed architecture, real-time traffic information is collected by utilizing a hybrid network to avoid congestion in an urban environment. Authors highlight that the proposed global path-planning algorithm not only improves the network traffic but also reduces the average vehicle travel cost by means of Lyapunov optimization.

Brennand et al. [72] present an ITS, in which RSUs are distributed throughout the city to detect and control congestions in their area of coverage. In the proposed system, RSUs are homogenously distributed based on the dimensions of the map and the maximum operating range of the RSUs. Vehicles create a message, including position, speed,

current path, and time they took to move through each road in their path, and send this information to the nearest RSU via single-hop communication, i.e., LTE or 3G. After receiving real-time information from the vehicles, each RSU builds a road graph model representing the region covered by their communication range. The roads on the graph are assigned a weight, inversely proportional to the vehicles' average speed traversing on them, and the congested roads are determined. After the creation of the weighted graph, each RSU re-routes the vehicles in their coverage area based on k shortest paths algorithm by using the Boltzman probability distribution. Authors claim that selecting a path from k shortest path alternatives contributed to load balancing. The simulations are carried out on a real and city-wide scenario by using the Veins framework, which integrates SUMO and OMNET++. It is found out that proposed distributed architecture can provide a 23% reduction in travel time, 9% reduction in fuel consumption, and 10% reduction in CO2 emission.

In [73], authors introduce SCORPION (System with COoperative Routing to imProve traffIc cONdition) as an ITS-based cooperative routing application. The proposed system consists of three main procedures: i) road network and communication model; ii) traffic condition classification; iii) cooperative routing. The road network is modeled as a directed and weighted graph that covers at least one RSU. The vehicles in the scenario periodically send their current position, speed, route, and destination information to the nearest RSU through a single-hop LTE or 4G communication link. Upon receiving new data from the vehicles, RSU computes the total number of vehicles and average speed for each road. Based on this computation, RSU updates the weight of each road, which is inversely proportional to the traffic condition on the road. In the next procedure, the traffic congestion on the roads is classified by employing the k-Nearest Neighbor (k-NN) algorithm trained by a synthetic dataset. Authors claim that k-NN is a simple technique, enables cooperative routing and avoids creating new congestions. In the cooperative rerouting part, global optimal routes are calculated for the vehicles that are potentially pass through the determined congested roads. A real road network is used to simulate the proposed model using the SUMO tool with TraCI. Authors compare the

SCORPION system with original traffic mobility trace (OVMT) without re-routing employed, and three other solutions from literature: DSP, RkSP and the solution proposed by Brennand et al. [72]. The cooperative re-routing procedure of SCORPION leads to a 17% reduction in average travel time and a 61% reduction in average stopping time. However, it also results in a 47% increase in the total traveled distance compared to the case without re-routing employed.

Authors of [74] propose DIVERT (A Distributed Vehicular Re-Routing System for Congestion Avoidance) that implements a hybrid architecture to avoid traffic congestion in urban scenarios. In the DIVERT system, vehicles communicate with the server to report traffic density data and receive global traffic data over cellular communication. By using VANETs, vehicles cooperate with the neighboring vehicles to determine local traffic density data, disseminate the traffic information received from the server and calculate alternative routes by considering the surrounding vehicles' future paths. Additionally, in the reporting of traffic density data to the server, a privacy-aware reporting mechanism is proposed to protect users' identities and origin-destination (OD) pairs. The simulation studies are conducted in two medium-size realistic road network scenarios by using the Veins framework. It is claimed that by offloading the route computation to vehicles, DIVERT reduces the network load on the server by 95% and achieves better scalability. Authors note that DIVERT is able to reduce the privacy leakage by up to 92% in the case when both distributed re-routing strategies, distributed Entropy-Based k Shortest Paths (dEBkSP) and distributed A* with Repulsion (dAR*), are employed.

Likewise, relying on a VANET environment, De Souza et al. [75] present a distributed and pro-active Traffic Management System, namely ICARUS (Improvement of traffic Condition through an Alerting and Re-routing System), to alleviate the traffic congestion problem in the urban areas. Upon receiving a traffic event (i.e., congestion detection, accident notification, and congestion prediction), the proposed system is utilized under two main phases: Information Dissemination and Re-routing. In the information dissemination phase, vehicles receiving a traffic event create an alert message including the set of roads that are likely to be affected by the traffic event. The alert message, char-

acterizing a congested area, is then disseminated to warn the vehicles within an Area of Interest (AoI) by using a vehicle-to-vehicle communication protocol. Authors define an area, namely sweet spot, such that only the vehicles inside this area continue performing data dissemination. According to the authors, the concept of sweet spot mitigates the broadcast storm problem and maximizes the coverage. Additionally, a desynchronization mechanism is implemented to minimize the number of packet collisions, such that if data transmission is scheduled at a time when the control channel (CCH) is active, an extra delay is added to swap the transmission to service channel (SCH). The re-routing phase of the ICARUS employs three different routing algorithms, which are Dijkstra, A* and Probabilistic k-Shortest Path, to compute alternative routes for the vehicles that will pass through the congested road section.

The Veins simulation framework, integrating Simulation of Urban Mobility (SUMO) and the Network Simulator OMNET++, is used to test the proposed VANET based re-routing scheme in a realistic road scenario. Regarding data dissemination, the proposed broadcast suppression mechanism is found to be effective in bandwidth usage. Authors further claim that their desynchronization mechanism resulted in the lowest number of packet collisions for all selected traffic densities, compared to other data dissemination protocols, i.e., Flooding. After the analysis of data dissemination, authors compare the routing performance of the ICARUS with three approaches from the literature (DSP, RkSP, and With routing) by simulating two congested scenarios caused by a high density of vehicles and a traffic accident respectively. Based on the simulation results, it is concluded that ICARUS is more efficient in pro-actively avoiding traffic congestion, reducing travel time, congestion time, fuel consumption, CO emissions, and maximizing the average speed of the vehicles in urban areas.

Zhang et al. [76] propose a fully distributed and infrastructure-less congestion avoidance system using VANETs, namely DIFTOS (Distributed Infrastructure-Free Traffic Optimization System). According to authors' claim, their work is the first completely infrastructure-free system aiming at avoiding congestion. The proposed system is modeled as a hierarchy of servers, such that vehicles located in the busy intersections act

as a server, called a virtual vehicular server (VVS). In every busy intersection area, vehicles collaboratively elect a vehicle to take the role of VVS based on a broadcast suppression scheme. The selected server vehicles keep the Road Reservations Matrix (RRM) containing the edge weights of the road network within the corresponding coverage. DOFTIS computes the link travel delay parameter for every road segment based on the flow speed. Additionally, a weight is assigned to every road segment based on the number of reserved positions on the road segment. The shortest path is found for each vehicle to their destination point based on the link travel delay values with a weight constraint. The simulation studies are performed on a real map representing a part of a real city y using the Veins framework, integrating SUMO and OMNET++. Authors note that DIFTOS is proven to be robust and scalable under varying traffic conditions.

In [77] authors propose a route suggestion protocol based on congestion awareness and travel time prediction. The system architecture composes vehicles equipped with IEEE 802.11p transceiver, vehicles without a wireless interface (non-equipped vehicles), RSUs deployed at intersections, and a central server unit. The proposed protocol utilizes two main procedures: traffic intensity monitoring and route suggestion. In the first phase, after receiving a route request message from vehicles, RSUs calculate congestion index (CI) for road segments as the ratio of an actual number of vehicles to a permissible number of vehicles. Authors consider both equipped and non-equipped vehicles to determine an actual number of vehicles on the road segments, which increases the accuracy of congestion index calculations. The route suggestion module calculates optimized routes for the vehicles, considering congestion and travel time prediction. When calculating travel times, the distraction factor is taken into account. To test the proposed protocol, authors use the Veins framework that combines SUMO and OMNET++. Based on the simulation studies carried out on an urban scenario, it is concluded that the proposed route suggestion protocol outperforms the existing approaches considering different metrics such as packet delivery ratio, throughput, travel time, and end-to-end delay.

An eco-friendly routing algorithm is presented in [78] based on VANETs with the aim of reducing both travel times and greenhouse gas emissions along the vehicles' routes.

The main components of the EcoTrec architecture are defined as a vehicle model, a road model and a traffic model. The vehicle model holds the important characteristics of each vehicle by using regularly provided inputs from the GPS sensors, speedometer, and accelerometer. The speed and position of each vehicle, determined by using the collected input data, are sent to the traffic model of other vehicles as VANET messages by relying on GeoRouting (GRP) protocol. The road model, stored in the central server, allows vehicles to query or update road-related information via RSUs using IEEE 802.11p. The traffic model component includes the overall traffic condition information of the road network based on average speed data. When vehicles receive updated road information, their optimum route is updated by the EcoTrec routing algorithm. The simulation studies are conducted on two real city networks by using the iTETRIS simulator that connects SUMO and NS3. Based on the simulation analysis carried out with varied scenarios, it is concluded that EcoTrec showed promising performance in terms of emissions and the percentage of vehicles that reached their destination.

In [79], authors introduce itsSAFE (Intelligent Transport Systems for improving SAfety and traFfic Efficiency) that provides optimum routes to vehicles with two objectives: i) minimizing traffic congestion and ii) minimizing unsafety level. The proposed scheme is modeled as an ITS system which comprises three main layers: i) traffic layer, ii) safety layer, and iii) processing layer. In the traffic layer, real-time traffic data is collected by using V2V and V2I communications in VANETs to have an accurate knowledge about traffic conditions on the road network. Relying on an official criminal database that includes information of crime events over the city, the safety layer assigns an unsafety level to each road. By exploiting data provided by traffic and safety layers, the processing layer detects bottlenecks and dangerous areas. With this information, itsSAFE periodically computes alternative routes with the highest traffic efficiency and with an unsafety level not greater than an acceptable threshold. The route planning modeled as Resource Constrained Shortest Path Problem (RCSP) and solutions are obtained by using a dynamic programming approach with a maximum tolerable unsafety level. The simulations are carried out by modeling a real road scenario in the SUMO framework.

Analyzing the simulation results, authors conclude that itsSAFE computes safe routes by providing a 55% improvement in overall traffic efficiency.

De Souza et al. [80] propose a non-deterministic multi-objective re-routing system, namely Safe and Sound (SNS), to optimize vehicles' routes in terms of both traffic efficiency and safety. In the proposed hybrid architecture, vehicles communicate with RSUs, 5G base stations, edge servers, and the remote cloud via cellular communication and vehicular networking. Authors partition the road network into equally sized subregions, in which a vehicle is chosen considering the number of vehicles in the vicinity to report the local traffic information. The traffic information is reported by the chosen vehicles to the server only if the traffic density is found to be higher than a threshold value. With these two requirements of the data reporting phase, it is aimed to reduce the number of transmissions and avoid network overhead. Upon receiving the traffic reports from the vehicles, the server determines the free-flow roads and estimates the traffic condition of other roads relying on the Greenshield model. Authors employ Recurrent Neural Network (RNN) as a deep learning technique to predict criminal densities and learn the safety risk dynamics of each neighborhood in advance. Vehicles then cooperatively compute their routes based on the Pareto set, which refers to a set of alternative routes optimizing both traffic efficiency and safety. Authors propose a rank-based approach for the cooperative re-routing in order to minimize the broadcast storm problem. By performing simulation studies on a real city network using the Veins framework, it is concluded that SNS provides improvements in terms of scalability, time complexity, and safety. It is noted that SNS provides a 30% reduction in the average safety risk for drivers by considering their preferences without degrading traffic efficiency.

3.2 Network-aware Route Planning Approaches

Among few research works we have found in the literature in this category, authors in [81] present a network-aware route planning approach by using a grid road network equipped with heterogeneous network access technologies such as WiFi, Bluetooth, and HiperLan. A smart vehicle passing through the defined geographic area is allowed to

set communication with these three types of access technologies installed in the middle of the road edges. The network-related cost assigned to each edge is defined in terms of the communication range and throughput of the technology deployed. By utilizing Dijkstra's algorithm on the built graph, a suitable route is found to provide always best connected service for the vehicles. Another connectivity-aware route planning tool is proposed in [82] using crowdsourced data from OpenStreetMap and OpenSignal. The road map data from OpenStreetMap is combined with a grid of boxes including signal strength data provided by OpenSignal. In the built graph, the portions of edges are assigned variable signal strength data. After constructing the graph, authors perform three types of route planning methods based on Dijkstra's shortest path algorithm. These routing methods are the classical shortest path, threshold-based route planning, and bounded degradation route planning. The threshold-based route planning method finds paths such that signal strength never drops below a certain threshold. In the bounded degradation route planning, paths are found that the time of experiencing weak signal is kept below an acceptable threshold value. Simulations are carried out on a medium-sized real map and the three routing methods are evaluated using the metrics of average RSSI, minimum RSSI, and path efficiency. In the end, it is noted that the proposed connectivity-aware routing tool provides considerable improvement in signal strength with a slight increase in physical path length.

When the presented approaches above are considered, there are few studies in the literature taking the communication network status information in the route planning mechanism. Moreover, we have not noticed a research study using the multi-objective evolutionary algorithms, i.e., NSGA-2, NSGA-3, for the dynamic routing of vehicles in the context of C-ITS. Thus, with the aim of addressing this research gap, in this work, we propose a modular framework that enables the collection of (near) real-time traffic and network status information from the road environment, and dynamically compute optimum routes for the vehicles, using the multi-objective evolutionary algorithm approach.

4 Overall Framework, Urban Scenario Model and Problem Formulation

This chapter outlines the system architecture and describes the main components of the overall framework. Additionally, the urban scenario model is described with its main constituents such as road network model, automated and connected vehicles, roadside units (RSUs), cellular base stations (BSs), and remote application server. In addition to describing the functionalities of the system components and the interaction among them via a vehicular network, the traffic-related and network-related data collection methods will be explained. The methodologies to estimate edge weights, corresponding to traffic condition and network quality, are described. We then formulate the vehicular re-routing process as a multi-objective optimization problem, which needs to be dynamically solved for each route requesting vehicle; and by considering both traffic-related and network-related metrics.

4.1 System Architecture and Overall Framework

The system architecture and the overall multi-objective route planning framework are presented in Figure 4.1. As a first step, the road network model and the corresponding road network graph are to be generated. One way to generate a SUMO-compatible road network model is to extract real-world map data. The real-world map data can be extracted from the OpenStreetMap [83], which provides data of roads, railways, rivers, etc., from all around the world. The map data from OpenStreetMap can be downloaded as an .osm file which can further be processed. If the manually selected area exceeds

the download limit, then the map data can not be downloaded directly from the OpenStreetMap. To download the map data of larger areas, the methodology identified in this study is to use the free geodata in the Geofabrik servers [84]. Geofabrik is specialized in OpenStreetMap services and provides current geodata of many regions (continents, countries, cities) of the world in various formats [84]. For instance, OpenStreetMap raw data can be downloaded from the public Geofabrik servers as a .osm.pbf file, which requires less memory [84]. The raw OpenStreetMap data files can then be processed by using various tools such as Osmium, PyOsmium, OSM4J, Osmconvert. In this work, the berlin-latest.osm.pbf raw data file is downloaded from the Geofabrik server [84] and the Osmconvert [85] tool is used to convert and process the raw OpenStreetMap data of Berlin. By providing the geographical borders (geocoordinates of southwestern and northeastern corners) and the berlin-latest.osm.pbf file as input, the .osm file representing a rectangular region can be obtained by using the Osmconvert tool.

A SUMO-compatible road network model can be generated from an .osm file by using the netconvert tool of SUMO [59]. Being a command line application, netconvert tool takes an existing .osm map data file as input and outputs a SUMO road network file in .net.xml file format. The road network data is encoded in .xml files in SUMO, which contains information regarding edges, lanes belonging to edges, traffic light logics, junctions, connections, etc. [59]. As noted in the documentation of SUMO [59], the OpenStreetMap data in .osm not only contains road data but also additional polygons such as buildings and rivers. This polygon data can be imported by using the polyconvert tool of SUMO and can then be visualized on sumo-gui [59]. Other than using real-world map data, the netgenerate tool of SUMO enables to generate abstract road network models, i.e., grid-like networks, spider-like networks, and random networks [59]. By this way, SUMO-compatible road network files can directly be generated in .net.xml format. The SUMO road networks can be imported to the visual network editor tool called netedit and can further be modified [59]. In this study, we have used the netedit tool to remove unconnected and isolated nodes and edges from the road network file, add traffic lights to some of the junctions, and modify the connections between lanes at some

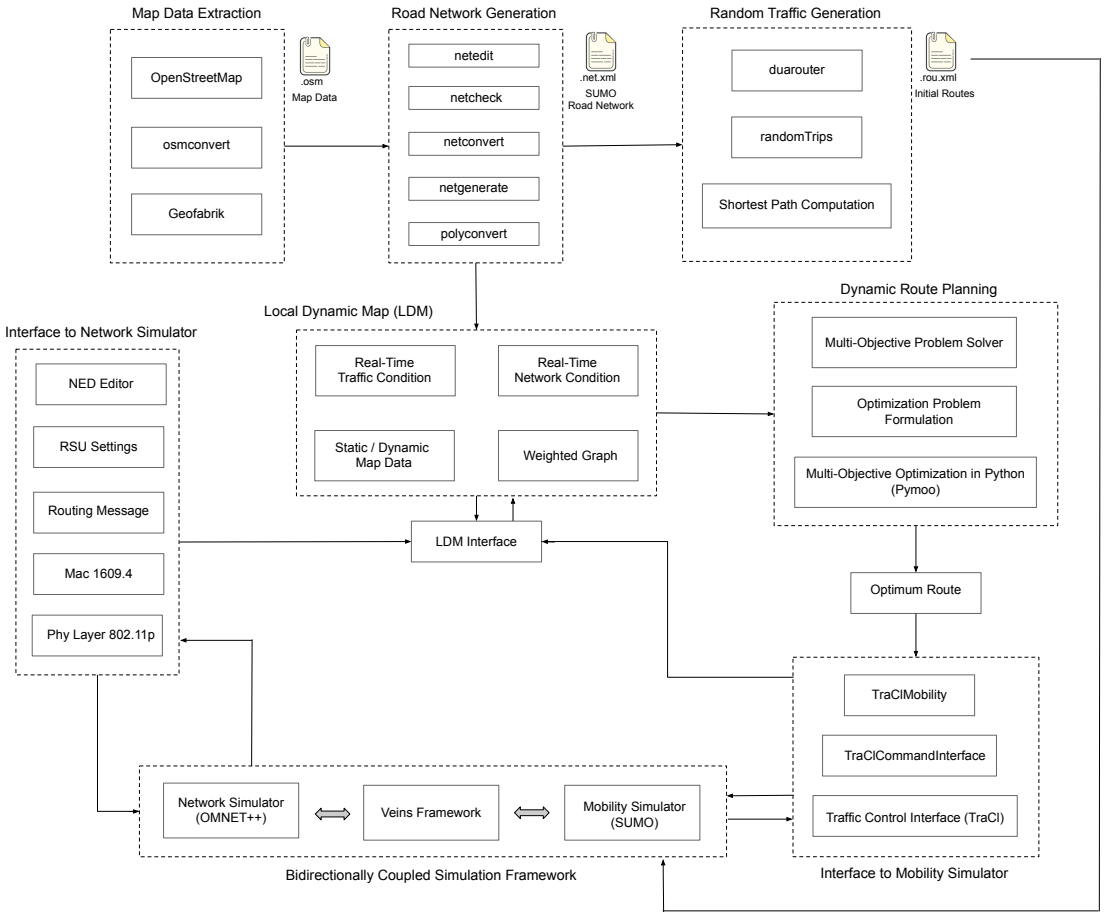

Figure 4.1: Depiction of the overall multi-objective route planning framework.

of the junctions. The netcheck.py tool of SUMO is used to check if the road network is fully connected or not [59]. In other words, netcheck.py allows to check if there is a valid route from every node to every other node in the road network [59]. The simulations and dynamic route optimization is run after it is ensured that the corresponding SUMO road network model is % 100 connected by using the netcheck.py tool.

Once a SUMO-compatible road network model is created, random trips are generated by using the randomTrips.py tool of SUMO [59]. The randomTrips.py tool takes the SUMO road network as input and generates random trips by randomly chosing source and destination edge pairs [59]. The randomly generated route set is saved in an XML file with a default name of trips.trips.xml [59]. While using the randomTrips.py tool, the start and end time can be specified for the trips and then random trips are evenly distributed between the specified time interval [59]. When employing the randomTrips.py tool for a road network, user is allowed to set different features for the generated set of random trips. For instance, as detailed in the SUMO documentation [59], the minimum straight-line distance between the source edge and the destination edge, the probability that source and destination edge are selected from the fringe of the network, arrival rate of the vehicles, departure attributes of the vehicles (depart lane, depart speed, depart position) can be specified when generating the random trips. After random trips are generated, the next step is to generate routes for the trips, which is realized by using the duarouter application of SUMO. The duarouter tool takes the road network and the random trips as input and computes routes by using a shortest-path algorithm, i.e., Dijkstra, A* [59]. The resulting routes are saved in an XML file with .rou.xml extension and are assigned as the initial routes of the vehicles involved in the scenario. Among the generated routes, each of which belongs to an individual vehicle, the ones that will be followed by routable vehicles (the vehicles that use dynamic route optimization) are specified in the .rou.xml file.

In addition to initializing the SUMO road network and the road traffic, the communication network simulation model is to be set and configured as well. First, the Network Description File (NED) can be edited in the editor area of the OMNET++ IDE by using

either Design mode or Source mode of the NED Editor [86]. In the Design mode, the simulated network can be graphically modeled using the Palette on the rght, whereas in the Source mode, the network source code is edited as text. By means of NED file, user describes the network, as a self-contained simulation model, which is made up of assembled simle modules and compound moduels [86]. The modules of the network model, i.e., RSU module, car module, are defined by NED editor. The network interface card of both RSU and car modules is specified as IEEE 802.11p for this study. As indicated in the OMNET++ documentation [86], when the simulation program is started the NED file is read at first, and then the configuration file omnetpp.ini is called. In the omnetpp.ini configuration file, the NED file, representing the network model to be simulated, is specified. The settings about the simulation, RSUs (i.e., location of the RSUs), network interface cards (i.e., IEEE 802.22p specific parameters), playground size are specified in the omnetpp.ini configuration file. This completes the communication network related settings before starting the simulation program.

A dynamic data storage entity, representing a Local Dynamic Map (LDM), is modeled as part of the remote application server component of the developed framework. By using the sumolib tool of SUMO, which contains a set of python modules to parse SUMO networks, an adjacency matrix model representing the road network is generated. The adjacency matrix, or the finite weighted graph model, is first instantiated based on the edge distances read from the SUMO road network file, and then, the generated weighted graph model is replicated for other objectives, i.e., travel time and congestion. A mechanism is implemented to update the entries of the matrices and retrieve the values from each data layer. Throughout the simulation run, the data layers are updated by the road traffic and communication network-related measuremensts.

To solve the dynamic route planning problem for the routable vehicles, we have integrated the Python-based optimization framework, namely Pymoo (Multi-Objective Optimization in Python) [87]. The modular structure of the Pymoo framework allows us to model many state-of-the-art optimization algorithms to solve any constrained, complex optimization problems. As the main external interface, Pymoo provides the

minimize function to solve the optimization problems. The minimize function of Pymoo takes arguments such as problem object that includes the optimization problem to be solved, algorithm object that stands for the optimization algorithm as the problem solver, and termination criterion object used to stop the algorithm [87]. Additionally, parameters to enable random seed, print out the outputs throughout the optimization process, save a snapshot of the algorithm at each iteration, and return the least infeasible solutions in the case that algorithm can not find a feasible solution, can be specified by the minimize function. By means of Pymoo, the optimization problem can be defined as a class by specifying a set of metadata including number of variables, number of objectives, number of constraints, lower and upper boundaries of the design space variables. These parameters are defined in the constructor method of the problem class and the evaluate method of the problem class is used to evaluate each of the generated solution alternative based on the defined objective functions and constraints. After defining the problem model, the algorithm as a problem solver is to be specified and customized. In this study, the aim was to handle the multi-objective route optimization problem for vehicles. Thus, we have mainly customized the multi-objective evolutionary algorithms such as Non-dominated Sorting Genetic Algorithm-2/3 (NSGA-2/3) to find out a set of optimal route solutions.

When an optimal solution is selected among the final solution set provided by the algorithm, the route representation of the solution is calculated based on the Priority-based Encoding methodology. First, the route node indices are found, and then the corresponding edge ids are determined. The SUMO-compatible route representation, the most optimal route calculated for the requesting vehicle, is set to the vehicle by using the mobility interfaces of the Veins framework, i.e., TraCI Command Interface. Throughout the simulation run, the mobility interfaces provided by the Veins framework are used to get the current traffic information from the vehicles in the scenario, which is then sent to the server application to update the dynamic data layers. Each vehicle also broadcasts its route information to the RSUs and vehicle nodes around. The broadcasted network packets are defined using the cPacket class which extends the cMessage class of

OMNET++. In this broadcasted message, the vehicle id, position and the current road id of the sending vehicle are encapsulated by using the Route Message API. Likewise, by means of the same API, the vehicle id and current road id of the sending vehicle, received signal strength, and the id of the receiving RSU are sent to the server application to make the required updates for the data layer corresponding to communication network. Thus, when the broadcasted network packets are successfully received by the RSUs, the dynamic data storage unit is updated accordingly with the extracted information.

At the lowest layer of the overall framework presented in this work, it is worth to note that there exists mobility and network simulation tools (SUMO and OMNET++), which are bidirectionally coupled via Veins framework. The microscopic mobility simulator SUMO simulates the vehicle movements based on a car following model, i.e., Krauß model. The network simulator OMNET++ handles the simulation of vehicular communication. Within this closed loop simulation setup, where online interaction is enabled, mobility and network simulators exchange data at runtime. In the following sections, we give an overview of the urban mobility scenario by introducing the C-ITS elements and formulate the vehicular route optimization problem.

4.2 Urban Scenario Model

This section describes the urban scenario model which consists of automated and connected vehicles, roadside units (RSUs) and/or base stations (BSs), and remote cloud servers that run vehicular applications, as depicted in Figure 4.2. Additionally, we detail the vehicular communication model that enables the wireless data exchange among the system nodes and present the road network model. The presented urban scenario model and the vehicular routing policy presented in this work are considered in the context of Cooperative Intelligent Transportation Systems (C-ITS), where various road elements (C-ITS stations) cooperate via different wireless access technologies, i.e., Dedicated Short Range Communication (DSRC), and 5G Cellular Communication. In the system model, vehicles with on-board units (OBUs) are able to communicate with other vehicles in proximity and with RSUs via vehicular networking. We further assume that

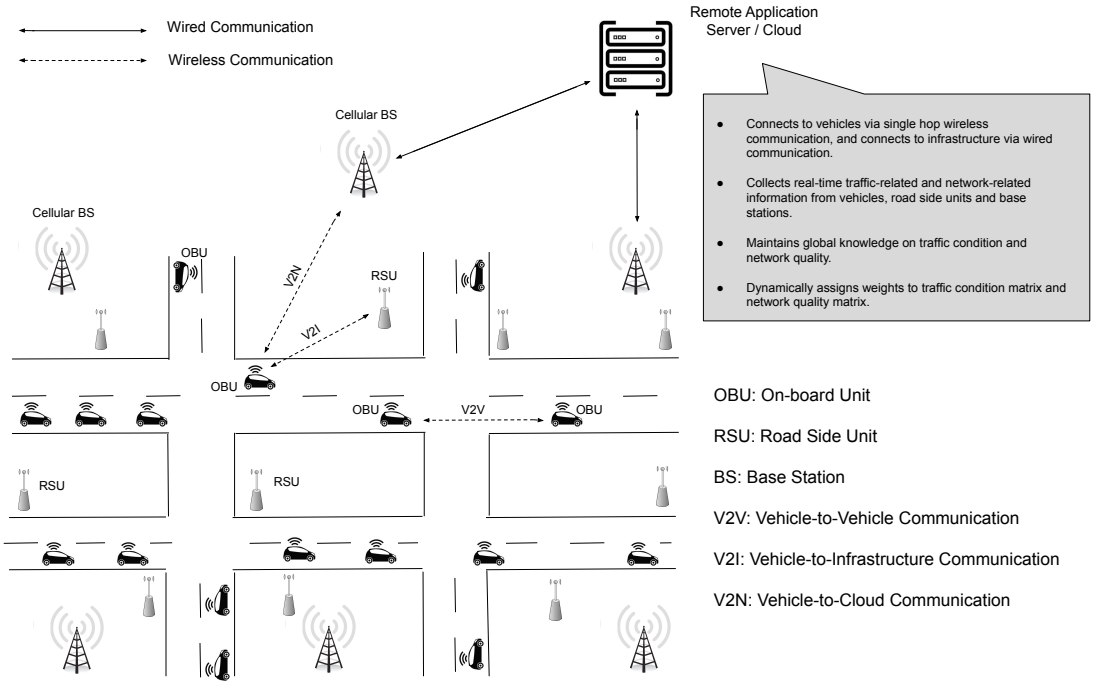

Figure 4.2: Depiction of an urban scenario comprising C-ITS elements such as automated and connected vehicles, wireless access points and remote application server.

there could be direct cellular communication between vehicles and base stations. In what follows next, we describe the main components that constitute the C-ITS model and take part in the proposed network-aware dynamic vehicle re-routing scheme.

Automated and Connected Vehicles: The automated and connected vehicles re-routed in our proposed system are to be equipped with a GPS-based navigation system, On-Board Units (OBUs), and wireless interfaces for both IEEE 802.11p standard and 4G-LTE/5G cellular communication. Likewise, vehicles comprising background traffic are also connected such that they communicate with RSUs or base stations modeled in the scenario via short-range or cellular communication mediums. All the vehicles in the urban scenario communicate with the RSUs and other vehicles in proximity via DSRC IEEE 802.11p-based short-range wireless communication technology, and with cellular base stations (or eNodeBs) via 4G-LTE/5G cellular communication. Vehicles periodically collect and send local traffic information to the nearest RSU. This traffic-related information includes vehicles' current location, speed, targeted destination, their currently followed route to reach their destination, the average speed of the road segment that they are moving on. These periodically sent traffic messages can be generated by using the measurements collected by their sensors and devices such as GPS receivers, and on-board diagnostics (OBD), or can be received from nearby vehicles via V2V communication. The periodically generated and sent traffic information is to be embedded in DSRC messages, standardized as Cooperative Awareness Message (CAM) or Basic Safety Message (BSM).

Road Side Unit (RSU): The roadside units (RSUs), equipped with DSRC-based wireless communication devices, are the gateways installed near the road infrastructure that communicates with the vehicles within their coverage via short-range wireless communication. RSUs can also be connected to each other via wired or wireless communication, to the cellular base stations, and the remote application server. The wireless ad hoc network established in the vehicular road environment is connected to the wired Internet via RSUs. Vehicles passing through their communication range exchange data with the RSUs via one-hop, short-range wireless communication. RSUs receive route requests and

collect local traffic information from the vehicles including vehicles' currently traveling route. With the collected information from the vehicles, RSUs determine local traffic conditions and congestion levels on the road segments associated with them. The route requests received from the vehicles are to be sent to the remote application server. The computed and recommended routes by the remote server can be delivered back to the vehicles through RSUs.

Base Stations (BS) / eNodeB: The cellular base stations (BSs) deployed throughout the city enable direct cellular communication between vehicles and the remote application server via 4G-LTE/5G communication. When the vehicles are out of the communication range of the RSUs, they can send traffic information to the remote cloud through base stations, or vehicles can receive traffic updates and route suggestions directly from the server via downlink cellular communication by means of the base stations.

Remote Application Server: The remote application server maintains up-to-date information about road traffic conditions and communication network quality of the overall road network graph. By collecting traffic and communication network-related information from all the connected vehicles through RSUs or BSs, the vehicular cloud node builds up a global knowledge of the urban scenario. With the collected information, it determines up-to-date values for the congestion level and network quality indicator associated with each road segment in the considered road network. The highly congested roads (bottlenecks) and the regions with low communication network quality are determined. This global, dynamically changing, and multi-layer graph structure refers to the Local Dynamic Map (LDM) concept introduced in Chapter 2. In addition to the traffic-related and network-related weights assigned to the edges (road segments), the application server maintains the current locations of the vehicles and the routes that they are moving on. Based on the collected information, the application server computes the globally optimum route solutions for the requesting vehicles. These route solutions computed by the server node are sent back to the vehicles and the vehicles are re-routed accordingly.

Road Network Model: The road network can be abstracted as a directed graph

Table 4.1: Notation: Road Network & Weighted Directed Graph

$G = (V, E)$	Directed graph representing the road network; Vertices V represent the junctions and Edges E correspond to links connecting them.
$R_d^{ij}, R_s^{ij}, E^{ij}$	Road, road segment and the edge connecting the junctions (nodes) i and j.
L^{ij}, D^{ij}	Physical length and distance of the road R_d^{ij}.
W_t^n	Waiting time at node n.
S_{avg}^{ij}	Average speed of the vehicles passing through road R_d^{ij}.
S_{max}^{ij}	Maximum speed limit of the road R_d^{ij}.
N_v^{ij}	Vehicle density on the road R_d^{ij}.
τ^{ij}	Traffic efficiency on the road R_d^{ij}.
T_c^{ij}	Current traffic congestion index on the road R_d^{ij}.
p_c^{ij}	Projected congestion contribution on to the road R_d^{ij} caused by only one vehicle v_m.
P_c^{ij}	Aggregated projected congestion contribution caused by all the vehicles registered to the road R_d^{ij}.
ϕ^{ij}	Communication network quality index assigned on the road R_d^{ij}.

$G = (V, E)$, where V represents the nodes and $E \subseteq VxV$ represents the edges connecting the nodes. While nodes of the graph correspond to intersections of the real road network, edges correspond to road segments connecting the intersections. The directed road network graph G is to be connected, which implies that there always exists a path connecting two nodes of the graph through other nodes. Each edge of the graph has a static traveled distance weight D^{ij}, and dynamically assigned weights of current congestion level T_c^{ij}, aggregated projected congestion contribution level P_c^{ij}, and communication network quality index Φ^{ij}. It should be noted that the last three weights associated with the edges are time-dependent and are to be updated based on the environmental changes, i.e., traffic and network status. Readers can refer to Table 4.1 for the notations regarding the road network model and the weighted directed graph representing the road network.

4.2.1 Determination of the Edge Weights on the Road Network Graph

This section explains how the edge weights assigned on the road network graph are determined. In this work, we have considered one static edge weight, which is the traveled distance along the roads taken from the OpenStreetMap data source. As dynamic edge weights, we consider the current traffic congestion, projected congestion contribution, and network quality index. Below, we introduce these dynamic edge weight metrics and describe how they are estimated throughout our simulation studies.

Estimation of Travel Time and Current Traffic Congestion on the Roads

The current traffic condition on the roads is estimated based on the periodically reported traffic information by the vehicles. Each vehicle in the scenario periodically sends traffic information to the server including vehicle ID, current road ID, current route, list of the mean speeds on the road segments that corresponding vehicle follows, list of the estimated travel times to traverse each road on the vehicle's current route, and the estimated remaining time to pass the current road. Based on the received information from a vehicle, the server-side application computes estimated travel time or the current

congestion levels on each road segment included in the current route of the corresponding vehicle. The same procedure is performed for every vehicle in the urban scenario that sends its periodic traffic information message.

To quantify the traffic condition on the roads, different congestion measures are defined in the literature as summarized in [88] under the categories of speed, travel time, delay, level of services, and congestion indices. In this study, we have implemented two methods to measure the current congestion level on the roads. As a first measure, we rely on Greenshield's model [80], [89]. Greenshield's model estimates the traffic condition on the roads by using the relation between average vehicle speed and vehicle density. First, the mean vehicle speed on a road is found by using Equation 4.1.

$$S_{avg}^{ij} = S_{max}^{ij} \ [1 - \frac{N_v^{ij}}{\frac{L^{ij}}{L^v} + \psi}] \tag{4.1}$$

where S_{max}^{ij} is the maximum speed limit on the road R_d^{ij} that connects the nodes i and j, N_v^{ij} represents the current vehicle density on the road, L^{ij} is the physical length of the road, L^v is the average length of a vehicle, and ψ represents the minimum distance gap between vehicles on the road. By using the estimated mean speed, the travel time (or traffic efficiency) on the road R_d^{ij} can be determined by using Equation 4.2.

$$\tau^{ij} = \frac{L^{ij}}{S_{avg}^{ij}} \tag{4.2}$$

Further, we define the current traffic congestion index on the road R_d^{ij} by using Equation 4.3.

$$T_c^{ij} = \frac{S_{max}^{ij} - S_{avg}^{ij}}{S_{max}^{ij}} \tag{4.3}$$

The other traffic congestion measure implemented in this work is the occupancy level on the roads, which is obtained by the SUMO mobility simulator during the simulation. Obviously, the total traffic efficiency on a route R^{sd} from current location s to destination location d is found by $\tau^{sd} = \sum_{ij \in R^{sd}} \tau^{ij}$. Likewise, the total current congestion of a route R^{sd} is given by $T_c^{sd} = \sum_{ij \in R^{sd}} T_c^{ij}$. One of the objectives considered in our optimization

problem is to minimize the total current congestion T_c^{sd} level on the routes of the vehicles. In order to normalize the routes' current congestion levels in between $[0, 1]$, we divide the total route cost by the number of road segments (edges) of the route, and the objective function regarding current congestion cost is defined in this way.

Estimation of Projected Traffic Congestion Contribution on the Roads

Inspired by [64], we have assigned an additional metric, namely projected congestion contribution, on the edges. This metric refers to the estimated, near-future congestion caused by each vehicle along their followed paths. More specifically, if a vehicle v^m is currently passing through edge E^{12} and is following the route $R^{1N} \rightarrow \{E^{12}, E^{23}, \cdots, E^{(N-1)N}\}$, then the corresponding vehicle would not only contribute to the current congestion level on its current edge E^{12}, but also contribute to the near-future congestion levels of upcoming edges $E^{23}, E^{34}, \cdots, E^{(N-1)N}$. Relying on this idea, we define a projected congestion contribution metric p_c^{ij} only estimated for the unvisited (to be visited along the route of a vehicle in the near future) edges of each vehicle.

The estimated projected congestion contribution values assigned to the next edges of a vehicle's route at time instant T_{curr} is depicted in Figure 4.3. As illustrated in Figure 4.3, the highest projected congestion contribution value is assigned to the next edge E^{23}, which is the closest edge to the current position of the vehicle. The assigned congestion contribution values decrease for the edges farther away from the vehicle, or in other words, for the ones closer to the destination node (n_6 in this case), s.t $p_c^{23} > p_c^{34} > p_c^{45} > p_c^{56}$. The idea behind this assertion is that a vehicle will pass through the edges that are farther away later. Thus, the congestion contribution values on these edges would be assigned lower than the ones assigned on the edges that will be traversed in the near future. In this work, the projected congestion contribution value is only estimated for the next edges that are not visited by the vehicles yet, hence, this metric is set to zero for the vehicles' current edges.

To be more specific, if we consider the case depicted in Figure 4.3, the vehicle is currently on the edge E^{12} and the projected congestion contribution value assigned on

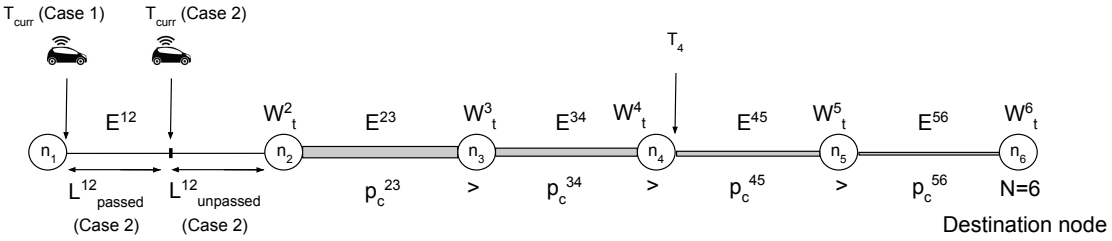

Figure 4.3: Projected congestion contribution along the edges of a vehicle route determined at time T_{curr}.

the current edge E^{12} would be zero, s.t $P_c^{12} = 0$ at the time instant T_{curr}. We estimate the projected congestion contribution caused by one vehicle on an upcoming edge based on the estimated time the vehicle enters the corresponding edge and the estimated time that vehicle reaches the destination location. If we specifically consider the example depicted in Figure 4.3, the projected congestion contribution p_c^{45} caused by the vehicle on the edge E^{45} can be calculated by using the estimated time T_4 the vehicle enters to the edge E^{45} and the estimated time T_6 the vehicle arrives its destination. While doing these calculations, we consider two cases as shown in Figure 4.3. In Case 1, the vehicle receives the suggested route from the remote application server when it just enters its current edge. In Case 2, the vehicle receives the calculated route when it is driving on the current edge E^{12}. We now illustrate how the projected congestion contribution value is calculated for the edge E^{45} by considering both cases in the following. Then, we form a generalized formulation.

Case 1: Vehicle receives the route when it just enters the current edge E^{12}:

First, in Equation 4.4 and Equation 4.5, we calculate the difference between the estimated time T_4 that vehicle enters to edge E^{45} and the current time T_{curr} at which the vehicle receives the suggested route solution from the server.

$$T_4 - T_{curr} = \frac{L^{12}}{S_{avg}^{12}} + W_t^2 + \frac{L^{23}}{S_{avg}^{23}} + W_t^3 + \frac{L^{34}}{S_{avg}^{34}} + W_t^4 \tag{4.4}$$

$$T_4 - T_{curr} = \sum_{\substack{i=1 \\ (j=i+1)}}^{4-1} [\frac{L^{ij}}{S^{ij}_{avg}} + W_t^{i+1}] \tag{4.5}$$

Based on Equation 4.4 and Equation 4.5, we can generalize the formulation for the difference between the estimated time T_u that vehicle enters to edge $E^{u(u+1)}$ and the current time T_{curr} as in Equation 4.6.

$$T_u - T_{curr} = \sum_{\substack{i=1 \\ (j=i+1)}}^{u-1} [\frac{L^{ij}}{S^{ij}_{avg}} + W_t^{i+1}], \qquad 2 \leq u < n_d \tag{4.6}$$

where n_d represents the destination node number.

Then, in Equation 4.7 and Equation 4.8, we calculate the difference between the estimated time T_6 that vehicle arrives at its destination and the current time T_{curr} at which the vehicle receives the suggested route solution from the server.

$$T_6 - T_{curr} = \frac{L^{12}}{S^{12}_{avg}} + W_t^2 + \frac{L^{23}}{S^{23}_{avg}} + W_t^3 + \frac{L^{34}}{S^{34}_{avg}} + W_t^4 + \frac{L^{45}}{S^{45}_{avg}} + W_t^5 + \frac{L^{56}}{S^{56}_{avg}} \tag{4.7}$$

$$T_6 - T_{curr} = \sum_{\substack{i=1 \\ (j=i+1)}}^{6-2} [\frac{L^{ij}}{S^{ij}_{avg}} + W_t^{i+1}] + \frac{L^{(6-1)6}}{S^{(6-1)6}_{avg}} \tag{4.8}$$

Based on Equation 4.7 and Equation 4.8, we can generalize the formulation for the difference between the estimated time T_{n_d} that vehicle arrives to the destination and the current time T_{curr} as in Equation 4.9.

$$T_{n_d} - T_{curr} = \sum_{\substack{i=1 \\ (j=i+1)}}^{n_d-2} [\frac{L^{ij}}{S^{ij}_{avg}} + W_t^{i+1}] + \frac{L^{(n_d-1)n_d}}{S^{(n_d-1)n_d}_{avg}} \tag{4.9}$$

where n_d represents the destination node number.

Case 2: Vehicle receives the route when it is driving through its current edge E^{12}:

First, in Equation 4.10 and Equation 4.11, we calculate the difference between the

estimated time T_4 that vehicle enters to edge E^{45} and the current time T_{curr} at which the vehicle receives the suggested route solution from the server.

$$T_4 - T_{curr} = T^{12}_{unpassed} + W^2_t + \frac{L^{23}}{S^{23}_{avg}} + W^3_t + \frac{L^{34}}{S^{34}_{avg}} + W^4_t \tag{4.10}$$

$$T_4 - T_{curr} = \frac{L^{12} - L^{12}_{passed}}{S^{12}_{avg}} + \sum_{\substack{i=2 \\ (j=i+1)}}^{4-1} [\frac{L^{ij}}{S^{ij}_{avg}} + W^i_t] + W^4_t \tag{4.11}$$

Based on Equation 4.10 and Equation 4.11, we can generalize the formulation for the difference between the estimated time T_u that vehicle enters to edge $E^{u(u+1)}$ and the current time T_{curr} as in Equation 4.12.

$$T_u - T_{curr} = \frac{L^{12} - L^{12}_{passed}}{S^{12}_{avg}} + \sum_{\substack{i=2 \\ (j=i+1)}}^{u-1} [\frac{L^{ij}}{S^{ij}_{avg}} + W^i_t] + W^u_t \tag{4.12}$$

Then, in Equation 4.13 and Equation 4.14, we calculate the difference between the estimated time T_6 that vehicle arrives at its destination and the current time T_{curr} at which the vehicle receives the suggested route solution from the server.

$$T_6 - T_{curr} = T^{12}_{unpassed} + W^2_t + \frac{L^{23}}{S^{23}_{avg}} + W^3_t + \frac{L^{34}}{S^{34}_{avg}} + W^4_t + \frac{L^{45}}{S^{45}_{avg}} + W^5_t + \frac{L^{56}}{S^{56}_{avg}} \tag{4.13}$$

$$T_6 - T_{curr} = \frac{L^{12} - L^{12}_{passed}}{S^{12}_{avg}} + \sum_{\substack{i=2 \\ (j=i+1)}}^{6-1} [\frac{L^{ij}}{S^{ij}_{avg}} + W^i_t] \tag{4.14}$$

Based on Equation 4.13 and Equation 4.14, we can generalize the formulation for the difference between the estimated time T_{n_d} that vehicle arrives to the destination and the current time T_{curr} as in Equation 4.15.

$$T_{n_d} - T_{curr} = \frac{L^{12} - L^{12}_{passed}}{S^{12}_{avg}} + \sum_{\substack{i=2 \\ (j=i+1)}}^{n_d-1} [\frac{L^{ij}}{S^{ij}_{avg}} + W^i_t] \tag{4.15}$$

where n_d represents the destination node number.

After the estimated time differences $T_4 - T_{curr}$ and $T_{n_d} - T_{curr}$ are determined for either Case 1 or Case 2, we define the projected congestion contribution caused by only one vehicle onto the edge E_{45} as in Equation 4.16.

$$p_c^{45} = \frac{(T_6 - T_{curr}) - (T_4 - T_{curr})}{(T_6 - T_{curr})} \tag{4.16}$$

In the generalized form, the projected congestion contribution caused by only one vehicle onto an edge included in the vehicle's route can be given as in Equation 4.17

$$p_c^{uv} = \frac{(T_{n_d} - T_{curr}) - (T_u - T_{curr})}{(T_{n_d} - T_{curr})} \tag{4.17}$$

This projected congestion value caused by each vehicle on to its next edges to be passed are updated when the vehicle receives a new route from the server, or when it moves to another edge. By this way, we maintain the updated projected congestion contribution values of every edge in the road network at the remote application server. The total projected congestion contribution level on an edge at a time instant T_k is determined by summing up the congestion contribution values caused by all the vehicles whose routes include that specific edge. Thus, if there are M vehicles whose routes contain the edge E^{ij} at time instant T_k, then the aggregated projected congestion constribution on this edge at this time is found by $P_{c_{T_k}}^{ij} = \sum_{m=1}^{M} p_{c_{T_k}}^{ij}$ (vehicles registered to the edge E^{ij}: $v^1, v^2, \cdots v^M$). The total projected congestion contribution associated to the currently followed route of a vehicle at time instant T_k is given by sum of the aggregated projected congestion contribution values assigned to each edge to be visited on that route, $P_{c_{T_k}}^{sd} = \sum_{\substack{i=2 \\ (j=i+1)}}^{N-1} P_{c_{T_k}}^{ij}$. One other optimization objective considered in this work is to minimize the total projected congestion contribution value $P_{c_{T_k}}^{sd}$ on the routes of the vehicles.

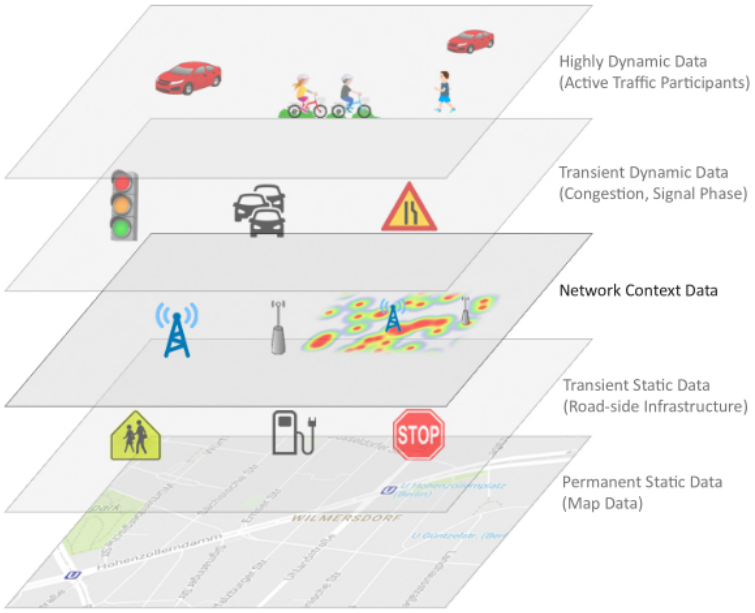

Figure 4.4: LDM enriched with a network context layer [2].

Estimation of Communication Network Quality on the Roads

Similar to the work presented by Varga et al. in [47], we extend the LDM by an additional layer that comprises Network Context (NC) Data (see Figure 4.4), which is collected by means of CAM messages. This additional network-related data layer comprises static and dynamic information regarding available access technologies.

In this study, we have modeled the urban scenario such that every connected vehicle periodically broadcasts its traffic information message, i.e., vehicle's current position, current road ID, to the nearest RSUs. The message type that the vehicle's traffic information is broadcasted is called WAVE Short Message (WSM) in the Vehicles in Network Simulation (Veins) framework. Once an RSU successfully receives a WSM message from a vehicle, it extracts two information from the message: (i) the road ID, corresponding to the current road ID of the vehicle that sends the message, and (ii) the received signal power in dBm (decibel-milliwatts). Then, the RSU sends its own ID, road ID, and received signal power (dBm) to the remote application server. Based on the sent information by RSU, the network layer of the dynamic map data structure is updated.

In more concrete terms, the entry of the network quality matrix corresponding to the received road ID is updated by a network quality index ϕ^{ij}, which is determined from the signal power level in dBm. We have identified maximum and minimum signal power levels as $-50\ dBm$ and $-110\ dBm$ respectively. When it is translated into network quality index metric ϕ between 0 and 1 ($[0,1]$); $-50\ dBm$ corresponds to $\phi = 1$, and $-110\ dBm$ signal power corresponds to $\phi = 0$. According to this relation, the received signal power in dBm is mapped to network quality index $\phi \in [0,1]$ at the server-side. The network-related edge weight γ^{ij} is defined as the inverse of the network quality index such that $\gamma^{ij} = \frac{1}{\phi^{ij}}$. Hence, the total network-related weight on a route from source s to destination d is defined as $\Gamma^{sd} = \sum_{\substack{i=1 \\ (j=i+1)}}^{N-1} \gamma^{ij}$, where N is the last node number in the route. The last optimization objective is to minimize the total network-related edge weight Γ^{sd} on the routes of the vehicles.

4.3 Multi-Objective Optimization Problem Formulation

A multi-objective optimization problem (MOOP) is mathematically formulated as in Equation 4.18 [90–94]:

$$\begin{aligned} \underset{x\in\Omega}{\text{minimize}} \quad & y = F(x) = (f_1(x), f_2(x), ..., f_m(x))^T \\ \text{s.t.} \quad & Ax \leq b_i \\ & g_j(x) \leq 0, \quad j = \{1, 2, ..., p\} \\ & h_k(x) = 0, \quad k = \{1, 2, ..., q\} \\ & lb \leq x_i \leq ub, \quad i = \{1, 2, ..., N_{var}\}, \end{aligned} \tag{4.18}$$

where the evaluation function $F(x) = (f_1(x), f_2(x), ..., f_m(x))^T$ represents a set of m objective functions $f_i : \Omega \to \mathbb{R}$ to be minimized. In Equation 4.18, $x = (x_1, x_2, ..., x_n) \in \Omega \subset \mathbb{R}^n$ is the vector of decision variables constituting the search space Ω and R^n is the n dimensional Euclidean space. The parameter $y = (y_1, y_2, ..., y_m) \in Y \subset \mathbb{R}^m$ represents the vector of objectives constituting the objective space Y. The $g_j(x) \leq 0, j =$

$\{1, 2, ..., p\}$ and $h_k(x) = 0, k = \{1, 2, ..., q\}$ are the inequality and equality constraint functions of the problem respectively. The lb and ub represent lower and upper boundaries of the variables, where N_{var} is the total number of variables in the problem model. Needless to say, the MOP can be also formulated as a maximization problem.

To substantiate, relying on the multi-objective optimization problem formulation presented in Equation 4.18, an instance of route planning problem can be defined by considering four objective functions such that evaluation function F becomes $F(x) = (f_1(x), f_2(x), f_3(x), f_4(x))^T$. For instance, the first optimization objective can be defined as minimizing the total traveled distance along the route, $f_1(x) = \sum_{\substack{i=1 \\ (j=i+1)}}^{N-1} D^{ij}$. The second optimization objective is to minimize the total current congestion level along the route, $f_2(x) = \sum_{\substack{i=1 \\ (j=i+1)}}^{N-1} T_c^{ij}$. The third optimization objective is to minimize the total aggregated projected congestion contribution level along the route, $f_3(x) = \sum_{\substack{i=1 \\ (j=i+1)}}^{N-1} P_c^{ij}$. The fourth optimization objective is to minimize the total network related edge weight along the route, $f_4(x) = \sum_{\substack{i=1 \\ (j=i+1)}}^{N-1} \gamma^{ij}$. We can define a constraint regarding the traveled distance metric. Let SP^{sd} is the shortest path distance from source location s to destination location d. Then, it can be stated that total distance of any route solution from s to d should not exceed $(1 + \delta)SP^{sd}$. This implies that $\hat{D^{sd}} \leq (1 + \delta)SP^{sd}$, and the constraint of this problem instance is defined as $g_1 = \hat{D^{sd}} - (1 + \delta)SP^{sd} \leq 0$. In the same way, different problem instances can be modeled with different objectives and constraints.

In many real-world engineering problems, the multiple objective functions f_i involved in the MOP are in conflict with each other, meaning that an improvement made in one of the objectives leads to deterioration on at least one of the other objectives [90] [95]. Due to that reason, a single global solution that optimizes all the objectives $f_i, i = 1, ..., m$ often does not exist. Rather, a set of best-compromising trade-off solutions can be found based on the well-known Pareto optimality concept [96]. The main goal in MOPs is to find out these best trade-off solutions, which are referred to as Pareto-optimal Solutions.

In the single-objective optimization problems, the goal is to minimize or maximize a single objective in the search space $Y \in \mathbb{R}$ and there exists only one optimum point on the

objective vector. Although there are several optimal solutions in the search space, they are all mapped to the same objective vector [97]. Unlike single-objective optimization problems, in the MOPs there exists multiple objective vectors in the objective space representing different trade-offs [97]. In this case, several optimal solutions in the search space might be mapped to different objective vectors, and the comparison of individual solutions becomes more complex for MOPs. In this regard, the well-known concept of Pareto dominance is used to compare the solutions $x \in \Omega$ and the objective vectors $y \in \mathbb{R}^m$ among each other. At this point, it is worth describing a set of definitions used in this context, such as Feasible solution set, Pareto dominance, Pareto-optimal solutions, Pareto-optimal front [90, 91, 93, 97].

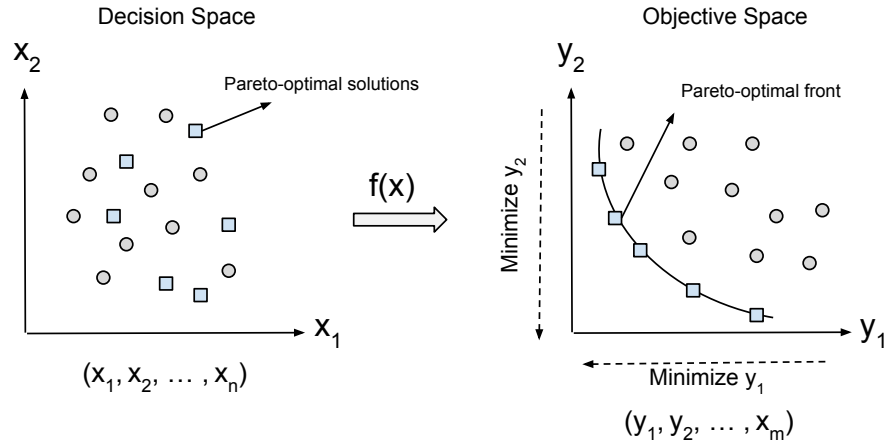

Figure 4.5: Illustration of the Pareto-optimal solutions in the decision space and Pareto front in the objective space.

Feasible solution set: The feasible solution set $X_f \subset \Omega$ is the set of solutions $x \in \Omega$ that satisfies all the constraints defined in the multi-objective optimization problem.

Pareto optimality: A feasible solution $x \in X_f$ is said to be Pareto optimal in the decision space Ω if $\nexists x^{'} \in \Omega$ such that $F(x^{'}) = (f_1(x^{'}), f_2(x^{'}), ..., f_m(x^{'}))$ dominates $F(x) = (f_1(x), f_2(x), ..., f_m(x))$.

Pareto dominance: Lets assume that x_1 and x_2 are two feasible solutions in the decision space. It is said that x_1 dominates x_2, denoted as $x_1 \prec x_2$ if $f_i(x_1) \leq f_i(x_2), \forall i \in$

$\{1, 2, ..., m\}$ and $\exists i \in \{1, 2, ..., m\}$ $s.t$ $f_i(x_1) < f_i(x_2)$.

Pareto-optimal solutions: A feasible solution $x^* \in X_f$ is said to be a Pareto-optimal solution if $\nexists x^{'} \in \Omega$ such that $F(x^{'}) \prec F(x^*)$. All the nondominated Pareto-optimal solutions $x^* \in \Omega$ comprises the Pareto set (PS), formally defined as $PS = \{x \in \Omega \mid x^{'} \in \Omega,\ F(x^{'}) \prec F(x)$. The Pareto-optimal solutions of a MOP are depicted on the left hand side of Figure 4.5 with blue rectangles.

Pareto-optimal front: The Pareto front is the image of the Pareto-optimal solutions (Pareto set) in the objective space. If $x \in \Omega$ is the set of Pareto-optimal solutions, the Pareto-front is denoted as $PF = \{F(x) |\ x \in \Omega\}$. The Pareto-optimal fronts of a MOP are depicted on the right-hand side of Figure 4.5 with grey circles.

5 Population-based Metaheuristic Approach for Vehicular Route Planning

Many real-world optimization problems are computationally complex, include multiple objectives to be satisfied simultaneously, and thus can not be always resolved in a feasible way by using the traditional, exact methods such as Integer Linear Programming (ILP). Especially, when the problem instance is relatively large, it becomes computationally expensive and infeasible to find out the globally optimum solution(s) by using the exact methods. The same fact holds true for the real-world vehicular route planning problem instances as one category of combinatorial optimization problems. In particular, this is valid for the cases that there are multiple objective functions to be minimized or maximized and there is a large road network model at hand. Considering the non-deterministic polynomial-time hard (NP-hard) vehicular route planning problem instances, the population-based metaheuristic algorithm approaches could be an effective solution alternative. Although it is not guaranteed to find globally optimum solution(s), the heuristic-based stochastic search methods are able to find sufficiently good solution(s) effectively. Regarding this, we adapt, implement and apply different versions of population-based evolutionary algorithm approaches to find optimal or near-optimal solution(s) for the vehicular route planning problem. This chapter introduces the evolutionary computation concept and its main principles in a broader view. Being one of the widely known and mainstream categories of the evolutionary computation paradigm, the Genetic Algorithm (GA) approach is illustrated over a single-objective route planning problem instance. In addition, we introduce the Multi-Objective Evolutionary Algorithm (MOEA) approach to resolve multi-objective vehicular route optimization problems and

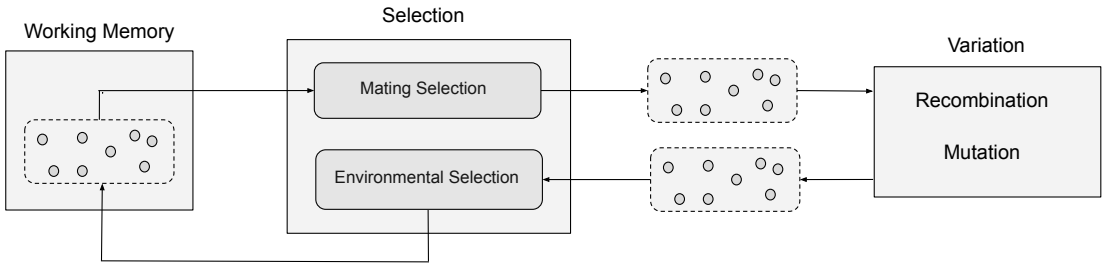

Figure 5.1: The stochastic search process in evolutionary algorithms [97].

we detail a state-of-the-art MOEA that we concern within this study.

5.1 Evolutionary Computation

The use of evolutionary computation techniques dates back to late 1950s, which are based on Darwin's biological evolution and natural selection theories. Since it has been originated in the 1950s, many different variants of the evolutionary computation paradigm are developed and applied to resolve complex real-world applications [98]. In 1960s-70s, three categories of evolutionary computation techniques are introduced: evolutionary programming, genetic algorithm, and evaluation strategies. Another class of evolutionary computation, namely genetic programming is devised in the 1990s. It has been highlighted that algorithms designed under the field of evolutionary computation are efficient, flexible, and sufficiently robust in solving various complex optimization problem types [99]. The methodologies designed in this domain, often referred to as evolutionary algorithms, imitate the concepts of the natural evolution process, i.e., selection, recombination, and mutation. In general terms, a set of solution candidates (individuals) are iteratively evolved towards the optimum set of solutions by using these bio-inspired algorithmic operators.

An overall stochastic search process utilized in the evolutionary algorithms is depicted in Figure 5.1 [97] and summarized in the Algorithm 1 [98] [100]. The evolutionary algorithms begin with an initial and often randomly generated population of candidate solutions. This initial population of solutions is then iteratively updated by using the

operators such as selection, recombination, and mutation throughout the evolutionary computation process (see Figure 5.1). In each generation of the algorithm, firstly, the current candidate solutions are evaluated based on particular objective function(s). Based on this evaluation, a scalar ranking value referred to as fitness value, is assigned to each individual in the population. This fitness value quantifies, to what extent a particular individual is fitting to be the solution of the optimization problem based on the objective function(s). The individuals with higher fitness values have more suitable genes, and closer to be the optimum solution to the problem. Thus, the individuals with high fitness values are given a higher chance to be selected for reproduction, that is, their features are transferred to the next generations. In this way, the less suitable solution alternatives are eliminated and the quality of the solution set is improved in every generation of the algorithm. The solution set is iteratively evolved until the maximum number of generations limit is reached or a predefined quality is satisfied by the solutions. In the end, an optimal or near-optimal solution set is obtained among which the best fitting solution(s) is/are selected to be the most optimum solution(s) for the problem. The following section illustrates the use of the genetic algorithm approach on a single objective route optimization problem, and in the following section, we give an overview of MOEAs that we have used to solve the multi-objective vehicular route optimization problem.

Algorithm 1: Evolutionary algorithm process.

Input : Initialize the population $P \leftarrow P_0$

Evaluate the initial population P_0

generation $g \leftarrow 0$

Output: Final population (set of solutions) P^*

while *termination criteria is not met* **do**

  Select parent individuals from P_i for recombination, and generate offspring Q_i

  Utilize mutation operator

  Evaluate $P_i \cup Q_i$ and select $P_i^{'}$

  Increment the generation number: $i = i + 1$

end

Output: $P \leftarrow P^*$

5.2 Single-Objective Route Planning using Genetic Algorithm Approach

The Genetic Algorithm approach is adapted and implemented to solve single-objective route planning problem. The objective of the route optimization problem is to minimize the total traveled distance by vehicle from its source location to its destination location. Formally, if we define an objective function $f : \Omega \subseteq \mathbb{R}^n \rightarrow R,\ \Omega \neq \emptyset,\ f^* = f(x^*) > -\infty$ is referred to as the globally minimum value $\Leftrightarrow \forall x \in \Omega : f(x^*) \leq f(x)$, for $x \in \Omega$, where x^* is the globally minimum solution [91].

This section introduces the Genetic Algorithm and illustrates how it is used to find the shortest-path route from a source node to a destination node on a directional graph representing a randomly generated road network. Prior to the multi-objective optimization problem model, which is the main concern of this study, we have analyzed the applicability of the Genetic Algorithm on finding the shortest paths with a single objective.

5.2.1 Genetic Algorithm

Genetic Algorithm is a widely used, effective metaheuristic technique to solve complex optimization and search space problems [101] [102]. The procedure followed in this algorithm mimics the genetic process of the biological organisms, which is based on the idea that over the generations, populations evolve based on the principle of natural selection or survival of the fittest according to Charles Darwin's theory [103]. Inspired by the concepts from evolutionary biology such as natural selection, crossover and mutation, the genetic algorithm iteratively evolves the solution candidates towards the optimum solution [104]. As mentioned in [103], in the natural behavior, the successfully performing and highly adaptive individuals survive well and would have a higher chance of mating. In other words, the successful and fit individuals have a relatively higher number of offsprings compared to poorly performing individuals [103]. Hence, the genes of the fit individuals spread to an increased number of offspring that will form the successive generations [103]. In this way, the combination of good characteristics from fit ancestors can produce superfit offspring having even higher fitness than their parents [103]. Simulating this biological evolution process representing natural selection, the genetic algorithm evaluates each individual in a population by using the objective function, assigns a fitness value to each individual, and generates new populations by selecting, mating, and mutating the individuals with better fitness value. According to the literature, the genetic algorithm is firstly described by John Holland in $1960s$ and developed by him and his colleagues in $1960s$ and $1970s$ [105]. It is used to resolve many real-world optimization problems in the fields of communication networks, image processing, route planning, and neural networks [101]. The procedure of the genetic algorithm and its operators, depicted in Figure 5.2, are well described in many sources in the literature [101] [102] [103] [104] [105] [106].

As seen in Figure 5.2, the population size N_{pop}, maximum number of generations N_{gen}, number of variables N_{var}, and the objective function $F(x)$ can be specified as input for the genetic algorithm. In most of the cases, the initial population is randomly generated, but as stated in [103], it can be generated by using a heuristic as well. The

random generation of the initial population allows the entire search space and wide range of solution possibilities to be considered [102]. The individuals (chromosomes), each of which corresponding to a solution candidate to the optimization problem, contain a number of genes that is equal to the number of variables N_{var} [101]. Then, each individual in the initial population is evaluated using the objective function such that a fitness value is computed and attributed to each one of the solution candidates. This fitness value assigned to individuals represents to what extent they can be an optimum solution for the problem. Based on the assigned fitness values, the individuals with higher fitness values are selected among the first generation. The selected individuals are crossed over to produce offspring for the new population representing the next generation. Thus, the newly generated population has a higher portion of good features compared to the previous generation [103]. By this way, the good features and characteristics, mixed and exchanged with other good features by means of the crossing-over operator, are spread to the subsequent generations [103]. Following the crossing-over operator, the mutation operator is applied that corresponds to making some minor random changes on the individuals of the newly generated population. The mutation operator allows the algorithm to explore the search space well and avoids being trapped in a local optimum [104]. The same procedure is repeated until the stopping criteria is satisfied, i.e., the maximum number of generations is reached or a solution in the last generation satisfies the predefined criteria [102].

Selection

The selection step determines which individuals of the current population will be involved in the formation of the next population. The parents from the current population are selected randomly such that they produce the children of the next generation as a result of the crossover operator. At this step, if more fitting individuals (ones with higher fitness value) are selected for crossing over, the genetic algorithm is to converge to the optimal solution in a shorter amount of time. However, while selecting the fit individuals, the premature convergence has to be avoided and the diversity of the solutions has to

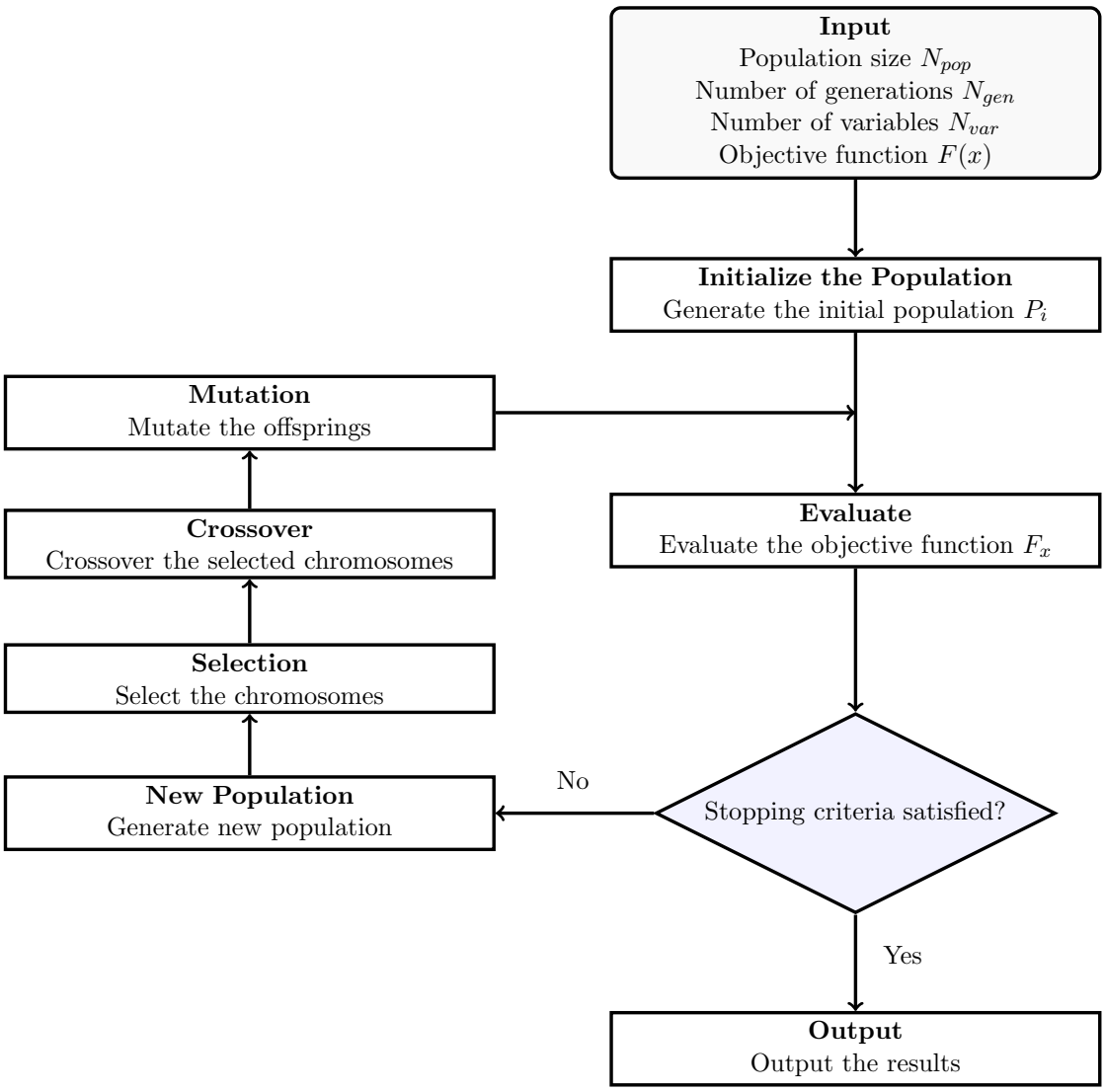

Figure 5.2: Flowchart representation of the Genetic Algorithm.

be maintained throughout the genetic algorithm procedure. There are different types of selection mechanism, i.e., uniform selection, roulette wheel selection, tournament selection, stochastic selection, etc., found in the literature.

In the uniform selection, the expectation and number of the parents are used to select mating parents to generate the next population [107]. The roulette wheel selection gives higher chance for the individuals with better fitness and gives minor chance for the ones with lower fitness to be selected. The selection of an individual is proportionate to its fitness value [108]. In this scheme, a pie chart is divided into several portions and the size of each portion is determined based on the fitness values of the individuals. In other words, each individual is associated with a portion of the pie chart based on its fitness value. Thus, the individuals with higher fitness values are associated with portions of bigger size, and the individuals with lower fitness values are associated with the small-size portions of the chart. In this way, if we think of that pie chart as a roulette wheel and when the roulette wheel is rotated, the fitter individuals would have a higher chance of being selected as their portions are bigger in size on the wheel. For instance, let us assume that there are four individuals in the current population with fitness values 10, 15, 25, and 50. To determine the individuals' chance of being selected, we can divide each one's fitness value to the total fitness value of 100. Then, the individuals would have 0.1, 0.15, 0.25, and 0.5 chance of being selected respectively. In the tournament selection method, an n number of individuals are randomly chosen from the current population, then the best fitting individual is selected out of this randomly chosen set of individuals for crossing over. The stochastic selection procedure associates current population's individuals to the segments of a line [107] [109]. Each individual is assigned to a segment with a length in proportion to the assigned individual's fitness value. After the segmentation, equally distanced pointers are placed over the line, i.e., if 10 individuals are to be selected, then 10 equally distanced pointers are placed over the line. As a result, the individuals whose segments are pointed out by the pointers are selected for crossing over.

Recombination (Crossover) Operator

The recombination, or crossover operator, refers to producing new solution candidates by mating the old ones in the search space. With the crossover operator, the characteristics and features of randomly selected parent chromosomes are merged and new solution candidates (offspring) are created for the next generation. The genes on the parent chromosomes are exchanged by following different crossover operators, which we detail a few of them in this section. By the crossover operator, it is aimed at creating more feasible and better solution candidates (offspring) by combining the beneficial genes and characteristics of the parent chromosomes.

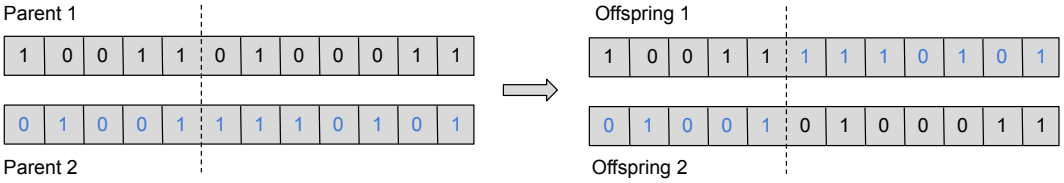

Figure 5.3: One-Point Crossover operator performed on a pair of binary encoded chromosomes to create two offspring.

Firstly, one-point crossover is the simplest and a widely applied crossover operator in genetic algorithm-based approaches. With this operator, a cut point is randomly selected for two chromosomes (parents) and the chromosome pairs are segmented into two portions. Then, as illustrated in Figure 5.3, the genes after the cut point are swapped and two offspring are created.

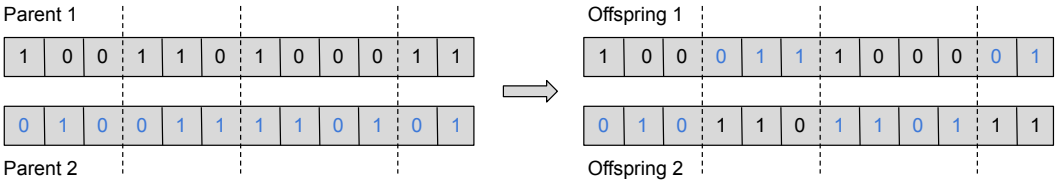

Figure 5.4: Multi-Point Crossover operator performed on a pair of binary encoded chromosomes to create two offspring.

In multi-point crossover operator, introduced by De Jong [110], multiple cut points are randomly selected for a pair of chromosomes and the segments split by these cut points

are alternatingly swapped between two chromosomes. As illustrated in Figure 5.4, two offspring are created as a result of the multi-point crossover operator performed on two binary encoded parent chromosomes.

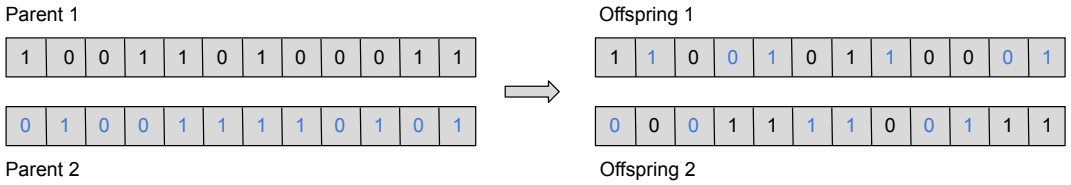

Figure 5.5: Uniform Crossover operator performed on a pair of binary encoded chromosomes to create two offspring.

In uniform crossover, the selected parent chromosomes are not split into segments to exchange parts of chromosomes. Rather, each one of the genes on the chromosomes is considered individually. While creating the offspring, a gene is swapped or not with a probability of 0.5. It is illustrated in Figure 5.5 that genes on the parents are swapped with 0.5 probability when creating the two offspring. In other words, as described in [111], the genes on the parent chromosomes are copied to the offspring based on an arbitrarily generated binary crossover mask. If the corresponding bit of the binary crossover mask is 1, the gene from the first parent is passed to the offspring. Otherwise, if the corresponding bit is 0, the gene from the second parent is copied to the offspring.

Given a real valued parent chromosome strings, parent 1 P_1 and parent 2 P_2, the intermediate crossover produces offspring using Equation 5.1 below, where O_i is the resulting offspring, n is the number of variables on a chromosome, and α is a uniform randomly chosen value over the interval $[-0.25, 1.25]$ [112].

$$O_i = P_i^1 + \alpha\ (P_i^2 - P_i^1), \qquad i = 1, 2, \cdots, n \tag{5.1}$$

The heuristic crossover operator makes use of the fitness values of parent chromosomes to decide on the search direction. The search is moved from the parent with a lower fitness value towards the parent with a better fitness value. In heuristic crossover

operator, the offspring are found by using Equation 5.2.

$$\begin{aligned} O_1 &= BP + \beta\ (BP - WP) \\ O_2 &= BP, \end{aligned} \tag{5.2}$$

where BP is the best parent, WP is the worst parent and β is a random number chosen in between $[0, 1]$.

Mutation Operator

The mutation operator is performed on a single offspring such that randomly selected genes on the offspring are exposed to a small change. The genes that undergo mutation are selected with the mutation probability rate p_m. More specifically, as noted in [113], if a randomly generated number within the interval $[0, 1]$ is greater than the predefined p_m, then a mutation is employed to the corresponding gene. Otherwise, the gene remains unchanged. The mutation probability rate should not be too small in order not to lose the genes that would have been beneficiary in the optimization process, and it should not be too high to avoid excessive perturbations that could take place on the individuals [101].

By means of the mutation operator, the lost genes from the population during the selection process can be replaced and retried, or the genes that are not present in the population can be provided [101]. Additionally, as noted in [114], it plays a key role in maintaining the genetic variation within the population by allowing the entire solution space to be searched for the most optimal solution. Thus, the mutation operator avoids the search algorithm to get trapped in local minima. As illustrated in Figure 5.6, two significant types of mutation operators are single gene mutation and multi gene mutation operators [101].

In the uniform mutation operator, two-step procedure is applied [107]. First, a fraction of genes (entries) of an individual is selected such that each selected gene has a mutation rate probability of being mutated. Then, in the second step, the entries to be mutated are replaced with a random value selected over the range of the corresponding

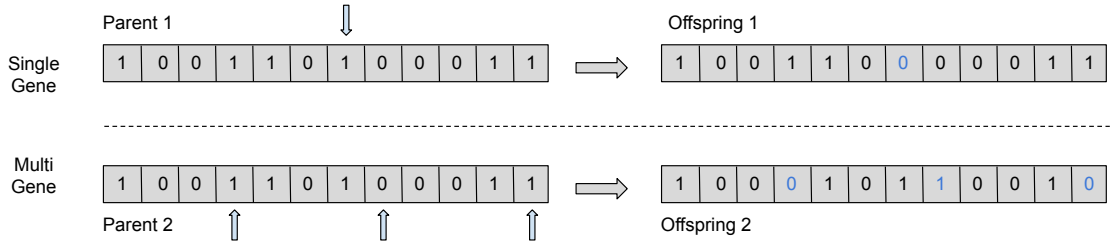

Figure 5.6: Single gene and multi gene mutation operators applied on a binary coded chromosome.

entry. The Gaussian mutation operator adds a random number from a Gaussian distribution with zero mean to the genes of an individual [107]. The standard deviation of the Gaussian distribution is determined by two parameters, scale and shrink. The scale parameter defines the standard deviation at the first generation, and the shrink parameter determines how the standard deviations shrink as the number of generations increases [107].

Priority-based Encoding Method

In the process of finding the shortest path by genetic algorithm, the encoding of an individual (chromosome) to a valid route (path) representation is critical [101] [115]. Being an effective solution encoding strategy, we have used the priority-based encoding scheme in this work to represent the given chromosomes as valid paths. The priority-based encoding method used with our genetic algorithm can be summarized as below [101] [115]:

1. Firstly, a route array with a length equal to the number of nodes in the road network is created. The first element of the route array is given as the current node ID. For instance, if we consider a road network with 200 nodes, then the size of the initialized route array would be $1x200$. If we assume that a vehicle is currently at node 1 and attempts to go to the destination node $n_d = 200$, then the first element of the route array is assigned as 1. This means the source node is set to $n_s = 1$.

2. An array maintaining the priority values is created with elements randomly generated in between a specified interval, i.e., $[1, 10]$. The indices of this priority array represent the node IDs.

3. Then, the neighbors of the source node are iterated by using the adjacency matrix $M_{200x200}$ corresponding to the road network with 200 nodes. During the iteration, if the searched node n_k is already in the route array, then set the priority value corresponding to this node n_k to minus infinity, $P_{nk} = -\infty$.

4. If a node n_k in the neighbor set of the source node n_s is found to be the destination node n_d, then the node n_k is appended at the end of the route array. As the destination node is found, the algorithm terminates at this point.

5. Otherwise, the priority values of the nodes within the neighbor set of the source node n_s are compared and the node with the highest priority value is appended as the next node to be visited at the end of the route array.

To illustrate the priority-based encoding method, we can consider the simple undirected network topology used in the article [115]. As depicted in Figure 5.7, the network graph has 6 nodes and 10 edges. In this example, it is assumed that a connected route or path is to be found from the source node n_1 to the destination node n_6. The priority values corresponding to each node are given by an array $P = [3, 5, 4, 6, 2, 1]$ [115] as seen in the figure. The first node is eventually n_1 is assigned as the first element of the route array. Then the priority values of the adjacent nodes to the current node n_1 are searched, which are n_2, n_3 and n_4 with priority values 5, 4 and 6 respectively. The one with the highest priority value 6 is the node n_4, so that n_4 is added to the route array as the next note to be visited. At this point, the current node is set to n_4 and the neighboring nodes of n_4 (n_1, n_3 and n_5) are searched. As n_1 is already added to the route array, it is not considered, and its priority value is set to $-\infty$. As the priority of n_3 is higher than that of n_5, n_3 is appended next to the route array. Among the neighbors of node n_3, the ones that are not added to the route array (n_2, n_5 and n_6) are considered. The node n_2 with the highest value is added to the route array. In the final step, as the node n_6

is the only unvisited node connected to the node n_2, n_6 is appended to the route array, which is also the destination node. In the end, the priority array $P = [3, 5, 4, 6, 2, 1]$ is represented as a route array as $R = [n_1, n_4, n_3, n_2, n_6]$ as seen in Figure 5.7.

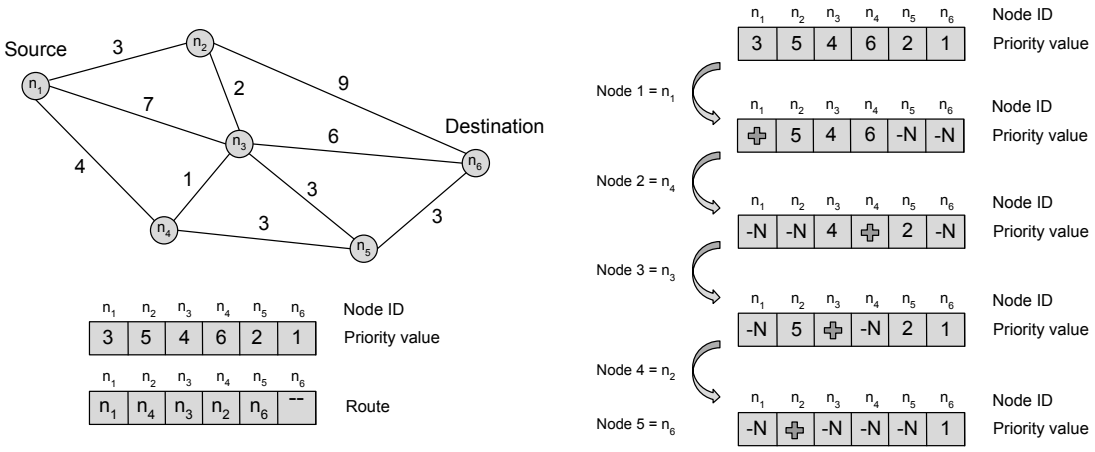

Figure 5.7: Priority based encoding for a network model [101] [115] [116].

5.2.2 Analysis of Single-Objective Route Optimization by using Genetic Algorithm

A random road network graph with 200 nodes and 560 edges is generated by using the netgenerate function of the mobility simulator SUMO (see Listing 5.1). The resulting .xml file RandomRoadNetwork.xml, representing the road network, is used to extract the distance of each road (edge) and the adjacency matrix is created. The diagonal and no-link entries are set to an extremely high value. We then run the Genetic Algorithm to optimize the shortest-path from the first node n_1 to the destination node n_{200}. We have used the priority-based encoding method to represent the randomly generated solutions as a route.

List of Listings 5.1: The command to generate random network in SUMO.

```
$ netgenerate --rand -o RandomRoadNetwork.net.xml --rand.iterations=200
```

The simulations are performed for different algorithm parameters, i.e., population size, selection type, crossover type, and mutation rate. The best score values are recorded for

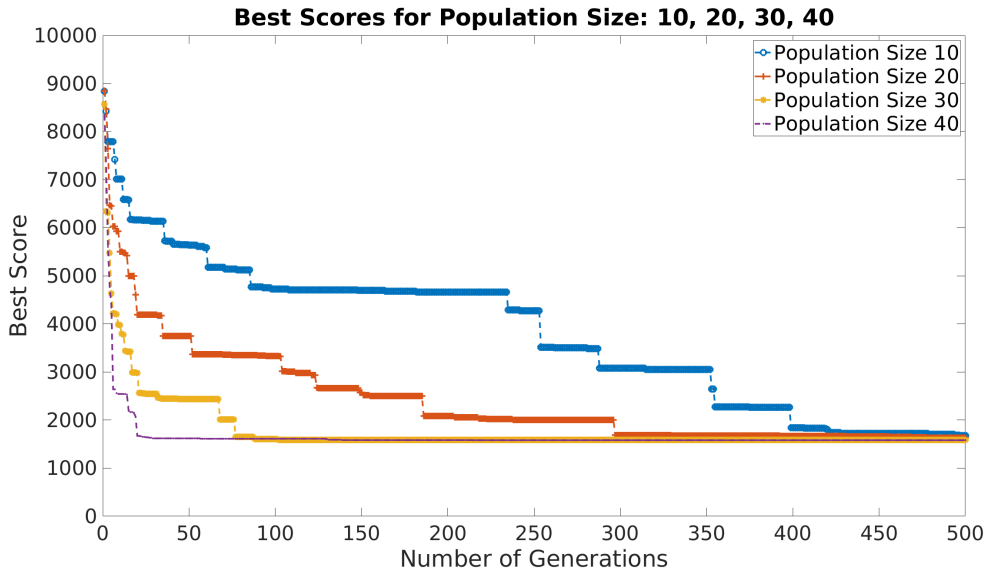

Figure 5.8: The best score plots for different population sizes, i.e., population size: 10, 20, 30, and 40.

every 20 iterations in each case and the mean best score values are plotted. As Figures 5.8, 5.9, 5.10, and 5.11 illustrate, the Genetic Algorithm effectively converges to the optimum solution as the number of generations increase. The algorithm first finds a very high number as the best score, and then, as the number of generations increases, the cost of the shortest-distance route is found closer to the expected value. In the following, we explain the four case studies we have performed to evaluate the operation of the Genetic Algorithm with different parameter settings.

Case study 1: In the first case study, we have run the Genetic Algorithm for different population sizes, i.e., population size: 10, 20, 30, and 40. As seen in Figure 5.8, the Genetic Algorithm converges to the optimum solution faster when the population size is set to be 40. The algorithm reaches the expected shortest-path route solution at approximately $25th$ generation. When the population size decreases from 40 to 10, the algorithm converges to the optimum solution in a longer time. As seen in Figure 5.8, when the population size is 10, the optimum result is found at around generation 450.

Case study 2: In the second case study, we have run the Genetic Algorithm for different

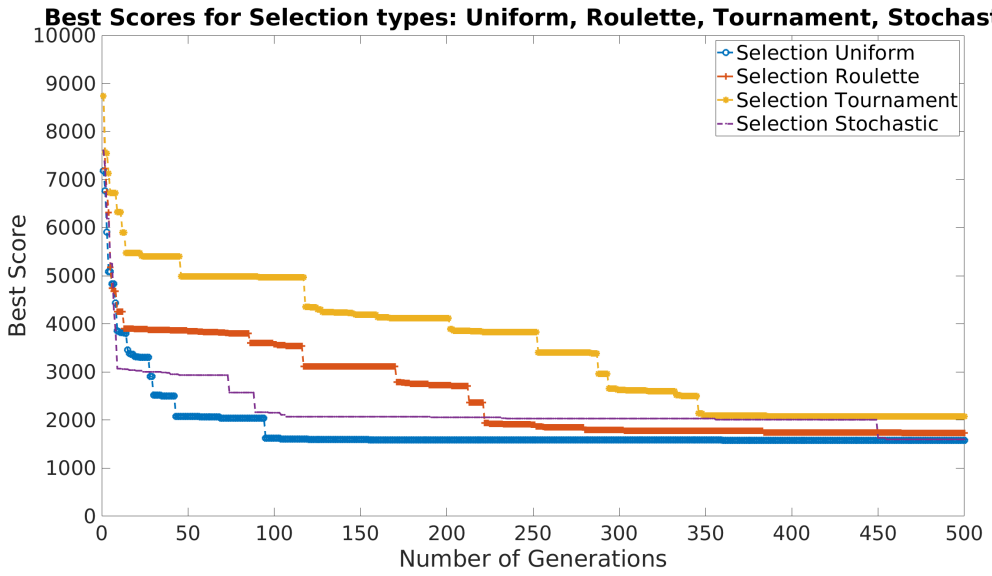

Figure 5.9: The best score plots for different selection types, i.e., uniform selection, roulette wheel selection, tournament selection, and stochastic selection.

selection types, i.e., uniform selection, roulette wheel selection, tournament selection, and stochastic selection. As it is observed from Figure 5.9, the uniform selection provides the best performance and the fastest convergence. The algorithm finds the shortest-path route at around $100th$ generation when the selection type is selected as uniform. In the case of tournament selection, the genetic algorithm can not exactly reach the optimum solution even when the generation number is 500.

Case study 3: In the third case study, we have run the Genetic Algorithm for different crossover types, i.e., single-point crossover, two-point crossover, intermediate crossover, and heuristic crossover. As it is seen in Figure 5.10, in the case of two-point crossover, the algorithm shows a promising performance as it converges to the optimum solution when the generation number is around 125. When the crossover type is set to heuristic, the algorithm does not show good performance compared to the other three options, as the best result is reached at around generation 470.

Case study 4: In the fourth case study, we have run the Genetic Algorithm for different mutation rates, i.e., mutation rate: 0.01, 0.08, 0.1, and 0.3. As Figure 5.11 illustrates,

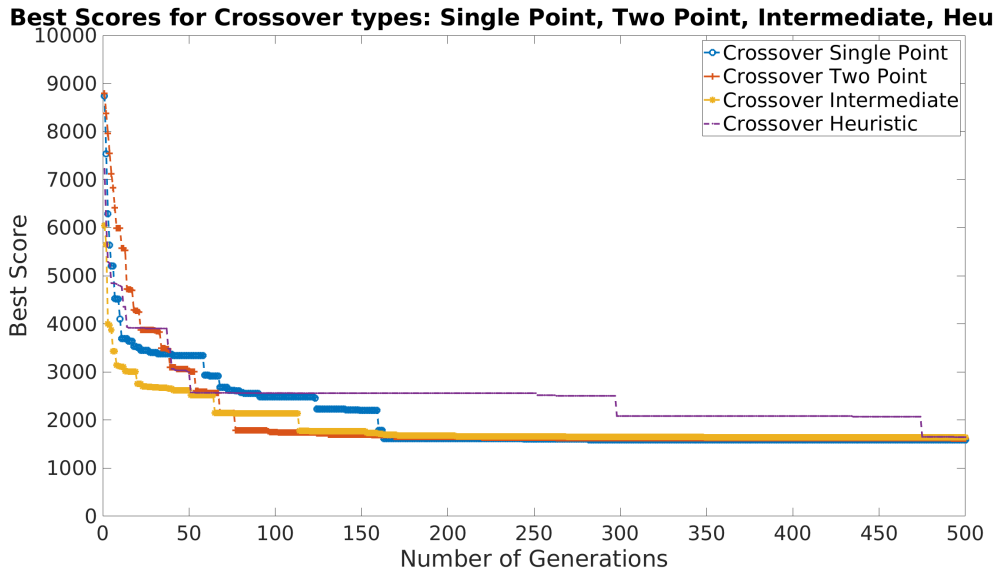

Figure 5.10: The best score plots for different crossover types, i.e., single point crossover, two point crossover, intermediate crossover, and heuristic crossover.

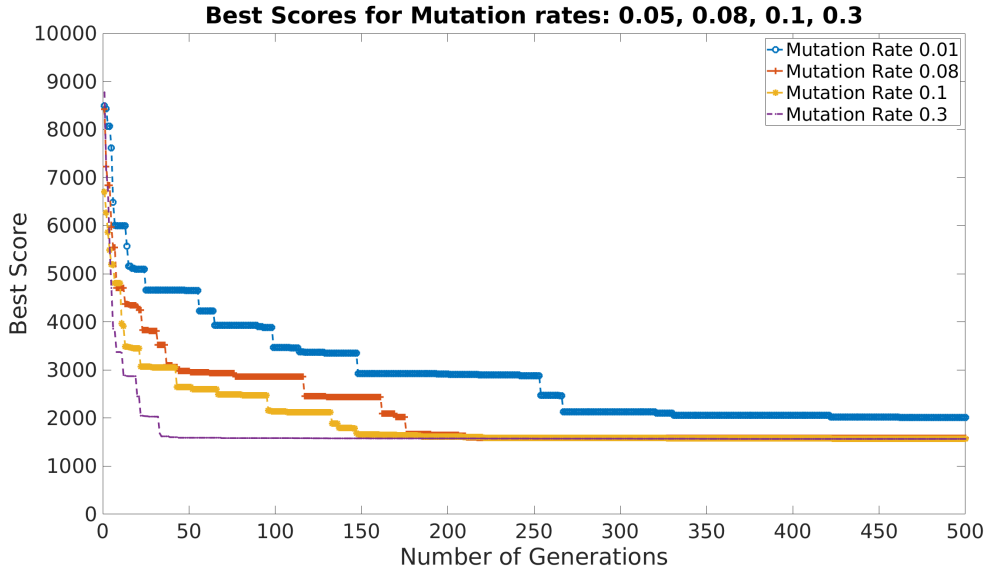

Figure 5.11: The best score plots for different mutation rates, i.e., mutation rate: 0.01, 0.08, 0.1, and 0.3.

the algorithm finds out the shortest-path route faster when the mutation rate is set to 0.3. When the mutation rate is given as 0.01, the optimum solution could not be reached even when the generation number is 500.

5.3 Multi-Objective Evolutionary Algorithm Approach for Vehicular Route Planning

The real-world combinatorial optimization problems usually require multiple and often conflicting objectives to be satisfied simultaneously. When the unique characteristics and requirements of the connected and automated vehicles are considered, the next-generation route planning frameworks need to be designed in a way that they can handle multiple objectives. In this regard, metaheuristic search techniques could be an effective approach to solve multi-objective vehicular route planning problems with constraints. Specifically, in this work, we have used population-based evolutionary algorithm techniques to search and find out trade-off solutions for the multi-objective route planning problem instances. The population-based evolutionary algorithms are suitable and robust techniques to handle multiple, conflicting objectives, highly complex, and large search spaces [97] [117]. Due to their population-based nature, the methods and algorithms in this category are able to provide a set of optimal and near-optimal solutions simultaneously in one simulation run [90]. Considering the requirement of finding multiple trade-off optimal solutions rather than finding a single optimal solution, this feature of population-based evolutionary algorithms is useful and convenient for the multi-objective optimization problems [117]. Especially, as highlighted by Zitzler et al. [97], the evolutionary algorithm approaches could be relevant in the cases that flexibility in the problem formulation is desired, we are interested in obtaining a Pareto-optimal set of solutions representing different trade-offs, and the high problem complexity does not allow the use of exact methods to solve the multi-objective optimization problem. In addition, the use of population-based evolutionary algorithms requires fewer domain information for the problem when compared to classical methods as highlighted in [93]

[118]. This section introduces a widely used, effective category of evolutionary search techniques, namely Multi-Objective Evolutionary Algorithms (MOEA), as a solution approach for the vehicular route planning problem instances concerned in this study.

There has been considerable interest in multi-objective evolutionary algorithms when dealing with multi-objective, real-world optimization problems since the mid-1980s [97]. Several variants of MOEAs have been designed, proposed, and used since David Schaffer's work published in 1985 that introduces the Vector Evaluated Genetic Algorithm (VEGA) approach [119]. The methods and algorithms proposed under this category are applied for solving both continuous and combinatorial multi-objective optimization problems in various engineering fields [90]. The algorithms classified under this category combine the basic principles of evolutionary computation and the classical multi-criteria decision-making procedure [97]. As mentioned by Zitzler [97], rather than aggregating multiple objectives into a single composite objective function, the evolutionary computation techniques used to solve multi-objective optimization problems mostly rely on the Pareto dominance concept introduced by Goldberg in 1989. Most of the MOEAs make use of the domination concept to compare the solutions in the presence of multiple objectives and aim at finding the non-dominated, trade-off optimal set of solutions in the end [117]. At this point, it can be noted that there are four primary goals in MOEAs [91] [97] [120]: (i) the nondominated set of points in the objective space and their corresponding solution points in the decision space need to be preserved throughout the MOEA process, (ii) the search process in the evolutionary algorithm needs to be steered towards the Pareto-optimal region and towards the Pareto-optimal front, (iii) the approximated set of optimal solutions obtained as a result of the evolutionary computation have to be diverse, and (iv) the algorithm has to provide a sufficient but also a limited number of Pareto points to the user for decision making. Among the resulting trade-off optimal solutions, it can not be said that a solution is better than the other optimal solutions without providing any further information [117]. In an ideal case, all the resulting optimal or near-optimal solutions are equally important, and the user makes use of high-level qualitative information to select one among them [117].

5.3.1 Non-Dominated Sorting Genetic Algorithm-2 (NSGA-2) Adapted to Vehicular Route Optimization

In this study, we have adapted the well-known and widely used multi-objective evolutionary algorithm to dynamically optimize the routes of vehicles, namely Non-Dominated Sorting Genetic Algorithm 2 (NSGA-2) proposed by Dep et al. [121]. The NSGA-2 algorithm is proposed by Dep et al. as an improved version of its predecessor Non-Dominated Sorting Genetic Algorithm (NSGA) [122]. The authors in [121] argue that there are three main drawbacks of the previous non-dominated sorting based genetic algorithm approach, which are noted as (i) high computational complexity resulting from nondominated sorting process, (ii) lack of elitism, and (iii) the need for specifying the sharing parameter σ_{share}. With the NSGA-2 algorithm, authors claim that computation complexity of the non-dominated sorting process is reduced from $O(MN^3)$ to $O(MN^2)$, where M is the number of objectives in the problem and N stands for the population size. The NSGA-2 algorithm combines the previous and the current population for the non-dominated sorting process. This way, the elitism is ensured by maintaining the most optimal solutions at every generation and a better convergence to the non-dominated front is achieved. As authors note, the diversity among solutions is maintained by crowding distance measure, which does not require to preset a parameter, i.e., the sharing parameter in the NSGA algorithm. Based on the simulation results obtained by considering difficult optimization problem sets, Deb et al. states in [121] that NSGA-2 outperforms two contemporary MOEAs, namely Pareto-archived Evolution Strategy (PAES) and Strength-Pareto Evolutionary Algorithm (SPEA), regarding convergence to the true Pareto-optimal solutions and divergence of the found solutions.

As illustrated in Figure 5.12, the adapted NSGA-2 algorithm mainly follows two processes, namely non-dominated sorting and crowding distance sorting. Firstly, the parent population P_t is created that contains the randomly generated individuals (solutions). The individuals specific to our vehicle routing problem are lists of random numbers in between predefined lower and upper bounds, i.e., $-1000 \leq x \leq 1000$. The size n of the lists representing the individuals are set to be the number of nodes in the studied

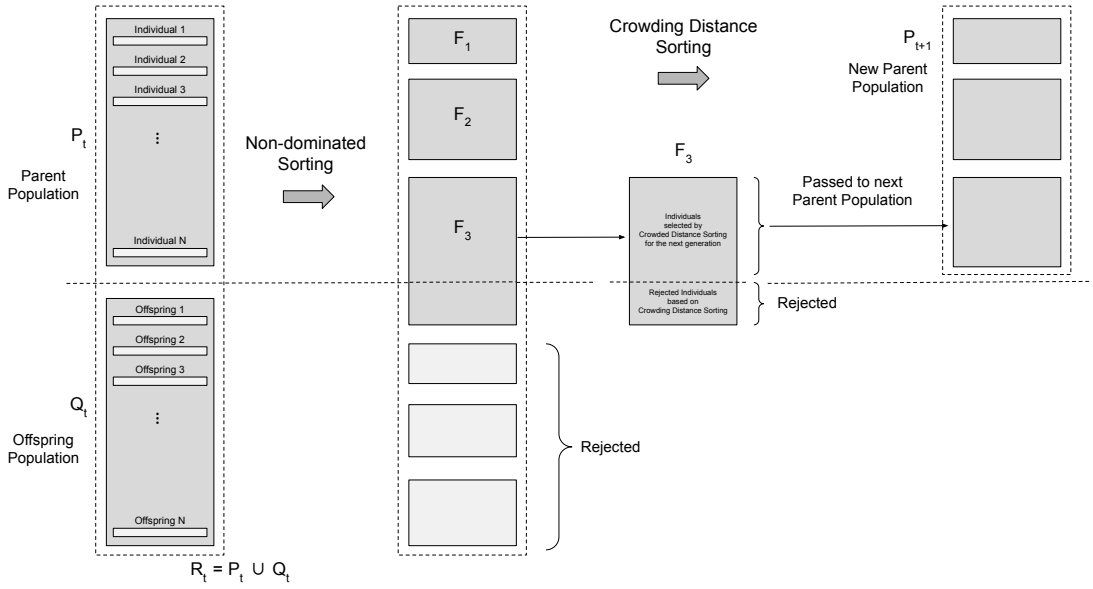

Figure 5.12: Illustration of the NSGA-2 algorithm.

road network model, and the population size is preset before starting the simulation, i.e., $N = 200$. Each and every individual (solution candidate) in the parent population P_t is assigned a fitness value based on the performance indicators defined in the problem. Specific to the vehicle route optimization problem considered in this study, these performance indicators can be defined as the total traveled distance, travel time, current congestion level, projected congestion level, network quality index of the route. These objectives can be varied by adding the necessary extensions to the simulation framework. It is important to note here that the individuals representing the solution candidates first converted into valid route representations by means of Priority-based Encoding methodology, and then their fitness values are computed using the objective functions. If a randomly generated solution candidate, containing numbers in between the preset boundaries, can not be converted into a valid route representation, its cost is eventually assigned to a very high value.

Next, the offspring population Q_t is generated out of parent population P_t by using predefined selection, crossover and mutation operators [87]. The generated combined population $R_t = P_t \cup Q_t$, with a size of twice as the size of the parent population, is used

for the non-dominated sorting process as illustrated in Figure 5.12. By means of the fast non-dominated sorting procedure, the individuals in the combined population R_t are grouped in different fronts F_1, F_2, F_3..., and are assigned a rank based on the front they belong to. The individuals in the lower number of fronts have a higher ranking value, i.e., an individual in the first front F_1 has a higher rank than an individual in the second front F_2. In order to classify the individuals into fronts, every individual p in the combined population R_t is compared to every other individual q, and it is pairwise checked if it dominates or is dominated by the other individual [121] [123]. While doing this pairwise checking, the number of individuals that dominate the individual p is counted and recorded, which is called the domination count n_p. Additionally, the list of individuals S_p that is dominated by the individual p is maintained. Taking this recorded information as a basis, the fast non-dominated sorting process is performed. The first front F_1 contains the individuals with domination count zero, and it should be noted that every individual in the first front has a list of individuals that it dominates. To determine the second front F_2, these lists of the individuals in the first front are iterated and whenever an individual q is found in these lists, the domination count of the corresponding individual n_q is subtracted one. Thus, as a result of this iteration, the individuals that are only dominated by the individuals in the first front would have zero domination count, which are placed in the second front F_2. The same iterative method is performed for the lists of the individuals comprising the second front, in order to determine the third front F_3 individuals. This procedure is repeated until all individuals in R_t belong to a front.

When the non-dominated sorting is completed, the individuals from the best fronts are chosen for the new parent population P_{t+1} until the original size of N is reached. When the size of the last front is higher than the remaining size of the new population, a set of individuals from the last front are selected based on the crowding distance sorting method. For instance, as illustrated in Figure 5.12, the size of front F_3 does not fit the remaining population size, and therefore, only a group of individuals are selected out of front F_3 for the next generation based on crowding distance sorting.

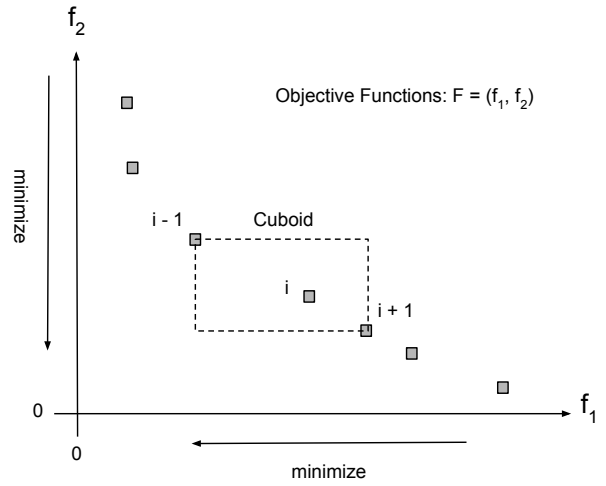

Figure 5.13: Illustration of the Crowding Distance Sorting.

By means of crowding distance sorting, the individuals in the front are sorted based on their Manhattan distance in the objective space [87]. Considering an intermediate individual i, two neighboring individuals are selected (corners of a cuboid, as seen in Figure 5.13). For each objective function in the problem, the distance of individual i is calculated as the normalized difference of the corresponding function values of two neighbor individuals $i-1$ and $i+1$ [121]. The total crowding distance of an individual i is then the sum of its distance values with respect to all objective functions. As mentioned in [121] [123], for each objective function, the individuals taking the minimum and the maximum function values are assigned a distance value of infinity. Then, the individuals with the highest crowding distance value are passed to the new parent population P_{t+1} from the last front. By this way, the individuals located in less crowded regions are selected and passed to the next generation, which eventually contributes to preserving

diversity [121]. The overall NSGA-2 algorithm is summarized in Algorithm 2.

Algorithm 2: Nondominated Sorting Genetic Algorithm 2 (NSGA-2).

Input : Size of the population N

Number of generations g

Output: Nondominated set of solutions.

Initialization: Randomly generate the initial population $R_t = P_t + Q_t$.
for $i \leftarrow 1$ **to** g **do**
    for *every solution in combined population,* $R_t = P_t + Q_t$ **do**
        Obtain the fronts by fast nondominated sorting, $F = (F_1, F_2, ...)$
        Calculate crowding distance
        Add solutions to the next generation starting from the first front until N solutions
    end
    Select solutions on the lower front with crowding distance
    Create the next generation by genetic operators: Selection, Recombination and Mutation.
end
The output is the nondominated set of solutions.

6 Implementation

6.1 Simulation Approaches for Vehicular Communication based Applications

This section gives an overview of the recent approaches and contributions for the simulation of Cooperative Intelligent Transport System (C-ITS) applications based on the Vehicle-to-Everything (V2X) communication paradigm. When the literature is examined, it is seen that there has been significant research effort in building up a simulation environment for the proper validation and assessment of various V2X-based applications or solution approaches, before their actual deployment in the field. Additionally, there is a number of comprehensive survey studies [124] [125] [126] [127] found in the literature, investigating the relevant research works by discussing their pros and cons. Below, we present a set of prominent research works in this context which we picked from the related literature.

In [128] authors carry out a simulation study to investigate the performance of delay-critical safety applications over commonly used DSRC wireless communication standard in a VANET model. The simulation study is performed as a two-phase process. First, authors perform a simulation study to evaluate the performance of the DSRC physical layer under a variety of vehicle speeds and multi-path delay spreads using Matlab. The physical layer performance of the DSRC standard is quantified by measuring the link bit error rate (BER). Based on the simulation results it is indicated that the DSRC standard tolerates very large delay spreads, however, it is sensitive to vehicle speeds and high mobility. In the second phase of their study, authors develop a simulation testbed

to evaluate how the collision avoidance safety applications are supported by DSRC based VANETs by considering realistic vehicular mobility models. As mentioned in the article, the road topology and the mobility model are ported from the CORSIM (CORridor SIMulator) vehicle traffic simulator which are then converted to the Qualnet simulator format. Authors conclude that the DSRC standard shows a promising latency performance for the time-critical collision avoidance applications, whereas the throughput is found to be moderate.

Authors in [129] emphasize the importance of using realistic mobility models for VANET simulations and introduce a tool in this regard, namely MOVE, which is built on top of the open-source simulator of SUMO. As detailed in the article, MOVE allows users to generate realistic vehicular mobility models that can be used by the widely used network simulators, such as ns-2 and Qualnet, to perform VANET simulations. Two main components of the developed tool are Map Editor and Vehicle Movement Editor. The former is used to generate maps either manually, automatically, or by importing from real-world map databases. The latter allows users to define vehicle movements manually by specifying the route properties, or automatically by specifying the vehicle flows. To evaluate the impacts of mobility models on VANET simulations, authors simulate and compare the performance of an ad-hoc routing protocol by using a realistic MOVE mobility model and random waypoint model, by taking the packet delivery ratio as a metric. Based on the simulation results, the packet delivery ratio is found to be considerably lower when a realistic MOVE mobility model is used compared to a simplified open field model.

Authors in [130] present the VanetMobiSim that provides extensions to a generic user mobility simulator previously developed, namely CanuMobiSim [131]. As noted in the article, VanetMobiSim extends the mobility model included in CanuMobiSim [131] by providing new features in terms of both macro-mobility and micro-mobility. From the macro-mobility perspective, VanetMobiSim includes the effects of points of interest on vehicle mobility. Additionally, it includes more detailed road structure characterization compared to CanuMobiSim. From the micro-mobility perspective, two additional models

are provided for the vehicles: (i) capability to handle intersections and (ii) possibility to utilize lane change and takeover maneuvers. With these additional features provided by VanetMobiSim, it is claimed by the authors that VanetMobiSim generates more realistic vehicular mobility traces to be used by telecommunication network simulators, which would result in more trustworthy VANET simulations.

Likewise, authors in [132] highlight the importance of using realistic mobility models for the evaluation of VANET applications via simulations and propose an integrated mobility and traffic model in this direction, namely STRAW (Street Random Waypoint). As detailed in the article, the proposed STRAW model makes use of a simple car-following model with traffic control. According to authors' claim, the introduced vehicular traffic mobility model enables the use of traffic information to dynamically adjust the routes of mobile nodes during the simulation run. Authors use two ad-hoc protocols, AODV (Ad-hoc On-Demand Distance Vector) and DSR (Dynamic Source Routing), and analyze and compare the performance of their proposed STRAW model with a classical Random Waypoint (RWP) Model. Based on the simulation carried out in three environments (an open field without streets, downtown Chicago and North End of Boston), it is concluded that the packet delivery ratio significantly varies when the STRAW model is used and when RWP is used for both ad-hoc routing algorithms. Additionally, when the runtime and memory overhead is investigated for the STRAW and RWP models, authors state that realistic and large-scale vehicular mobility can be successfully modeled in commodity hardware.

With the aim of providing a stable, easy-to-use, and realistic vehicular network simulator to the VANET community, authors in [133] design and present GrooveSim, which includes a variety of mobility, trip, communication, and traffic density models. It is stated that the GrooveSim allows to evaluate protocols by simulating thousands of vehicles across wide-scale road topologies. Five mode operation of GrooveSim covers actual on-road inter-vehicle communication, simulation of traffic networks with thousands of vehicles, visual playback of driving logs, hybrid simulation composed of real and simulated vehicles, and easy test-scenario generation as listed in the article [133]. Authors

use GrooveNet, a geographic broadcast protocol to evaluate the GrooveSim that supports two modes of messaging, diffusion and directed broadcast. Based on the simulation results, authors state that geographic broadcast routing is effective in delivering time-bounded messages over multi-hop communication.

The above-mentioned approaches rely on a two-step procedure that is utilized sequentially. First, as a preliminary step, the mobility traces are generated by means of a traffic simulator. Then, the generated mobility files are fed into a network simulator in an appropriate format. Afterwards, the network simulator is run based on this imported mobility data. This approach can be valid for the uni-directional scenarios where the vehicular mobility is not affected by the network situation, i.e. infotainment applications. However, for the scenarios in which the network status affects the mobility behavior of the vehicles, the simulation setup has to enable bi-directional data exchange between the mobility simulator and the network simulator. In other words, a closed-loop mechanism needs to be set up that allows mobility and network simulators to exchange data in the runtime of the simulation. In the following, we present the simulation frameworks modeled and developed in this direction. When the studied concept in this work is considered, the integrated and bidirectionally coupled simulation frameworks allowing online interaction in the runtime would be a more promising option. Thus, among the simulation setups mentioned below in this context, we chose to use the Veins (Vehicles in Network Simulation) framework for this work, which will be detailed in the following section.

Authors in [134] make extensions to the SWANS wireless network simulator to enable two-way communication between the mobility model and the networking model. With these extensions, feedback between the application layer and the mobility model is provided, which enables two-way communication. For instance, a customizable highway topology is built to support highway scenarios. As another extension, the mobility model of the SWANS is extended to enable vehicle control, such as acceleration, deceleration, and lane change. Additionally, the (probabilistic) Inter-Vehicle Geocast (IVG) broadcasting technique is implemented and an enhanced node model is included for the

addition of non-communicating vehicles, road-side units, and obstacles to a simulation. Finally, statistical and logging facilities are included in the extended framework for the reporting of simulation metrics. According to authors' claim, these additions to the SWANS network simulator are to allow for realistic VANET simulations of important safety and traffic information applications.

Another hybrid simulation framework, namely GrooveNet, is presented and detailed in [135] that supports vehicular communications. As highlighted by the authors, GrooveNet enables the communication between real and simulated vehicle nodes, such that vehicles in close proximity are able to exchange packets. Authors indicate that this feature provides rapid development as well as correctness and stress testing of vehicular network protocols. It enables the prototyping of test-beds for multi-hop communication on the road, as noted by the authors. Regarding scalability, according to authors' claim, GrooveNet allows running simulations consisting of thousands of virtual vehicles in any US city.

In [136], authors present their simulation architecture, namely Traffic and Network Simulation Environment (TraNS), targeting realistic simulation and evaluation of VANET based applications. As stated in the article, TraNS enables the coupling of SUMO, as the traffic simulator, and NS2, as the network simulator. Authors highlight the two distinct modes of operation provided by TraNS: network-centric mode and application-centric mode. In the network centric mode, the mobility traces are first generated by the traffic simulator prior to the network simulation. The mobility-related information is then parsed and translated into a format that can be used by the network simulator. On the other hand, in the application-centric mode of TraNS, the traffic and network simulators run simultaneously. This mode does not require the initial generation of mobility traces from the traffic simulator for the use of a network simulator. In this case, information exchange is enabled between both simulators in runtime, via a specific interface, called TraCI. In this way, the mobility of the individual vehicles can be controlled and manipulated by the network simulator in the runtime of the simulation.

In [70], authors couple the vehicular mobility simulator MobiDense with the network

simulator Qualnet, which continually exchange information at each step of the simulation. As authors detail in the article, by combining topology and traffic flow information, MobiDense generates mobility traces for the vehicles. It allows the dynamic update of the road weights, provides detailed street information, and routes the vehicles based on the topology and the estimated traffic conditions. Qualnet simulator, using the vehicles' positions and the streets' traversal times provided by the MobiDense, disseminates the information based on a gossip-based ad-hoc routing scheme. Authors test and evaluate their proposed decentralized, dynamic vehicular routing system, namely CATE (Computer-Assisted Traveling Environment), by using this coupled mobility-network simulator setup. As detailed in the article, in the designed CATE system, each vehicle acts as a traffic probe and creates traffic samples (linkID, delay, timeStamp, carID) every time it exits a road segment. Then those traffic samples are broadcasted to neighboring vehicles based on a sample selection algorithm by using a utility function representing the effectiveness of each sample. Each vehicle then estimates the traffic conditions based on the received samples and dynamically reroutes itself by using Dijkstra shortest path algorithm run on a weighted graph. Based on the simulations, authors conclude that their decentralized routing model can reduce traffic congestion in a realistic scenario.

Vehicles In Network Simulation (Veins) [137, 138] is a widely used hybrid simulation framework that enables bi-directional coupling of road traffic simulator (SUMO) and network simulator (OMNET++) to test and evaluate various V2X-based solution approaches. Authors in [137] underline the necessity of bi-directionally coupled simulators for realistic modeling and evaluation of VANET scenarios, by proposing their Veins framework. In the paper, the bi-directional coupling and the information exchange between the SUMO and OMNET++ simulators are described in detail. To demonstrate the viability of the developed Veins framework, two sets of experiments are carried out. In one set of experiments, authors use a centralized inter-vehicle communication (IVC) model, where vehicles maintain a TCP connection to the central server to exchange messages. In the second set of experiments, a decentralized and self-organizing model is used for the information dissemination among vehicles. These two sets of experiments are run

by considering the Manhattan grid scenario and the street map of Erlangen-Germany. With the presented simulation results, authors indicate that the IVC significantly alleviates the negative impact of an artificial accident on travel times. Additionally, it is found out that the simulations that included an artificial traffic incident, but no IVC, recorded a larger amount of Co2 emission. In the end, authors highlight that with their Veins framework, enabling the bidirectional coupling between two state-of-the-art simulators, the impact of IVC on the road traffic or environment can be accurately investigated.

Authors of [139] test and evaluate their intelligent routing approach, based on VANETs, by using the Veins framework as the mediating interface between SUMO and OMNET++. The SUMO is used to generate synthetic traffic demand by assigning random source and destination points. By means of the Veins framework, the vehicular nodes in the SUMO are mapped with the network elements in OMNET++. In the simulation scenario, authors periodically generate accident situations over traffic regions. Once the accident is simulated, the vehicle involved in the incident broadcasts a notification message to the neighboring vehicles. Upon receiving the message, the neighboring vehicles act based on their location, accident location, and their intended travel path. The simulations are run on the study regions from Colombo and Kandy cities in Sri Lanke, by enabling and disabling the re-routing component of the vehicles. Based on the simulation results, authors conclude that the trip duration and waiting time of vehicles significantly get reduced when the re-routing component is enabled.

Authors in [140] build up an integrated simulation framework, based on High Level Architecture (HLA), by combining VISSIM, NS-2, and connected vehicle (CV) application simulator. As indicated in the article, the HLA architecture provided a standard interface and simulation logic for the flexible integration of different simulation tools. With the support of the real-time infrastructure (RTI) component, a valid simulation logic and coordination between the VISSIM and Ns-3 is ensured. Authors specifically underline the advantages provided by the HLA-based architecture, such as promoting interoperability and reusability of simulation components. The proposed simulation framework is validated by demonstrating two case studies: i) speed guidance application

at intersections and ii) intersection collision warning.

In the same direction, authors in [141] combine VISSIM and NS-3 by using Matlab as a defacto RTI for the purpose of simulating V2X operations. The inbuilt Waypoint Mobility Model of NS-3 is used for the simulations such that the mobility information from VISSIM is correctly reflected in NS-3 in an on-line manner. As detailed in the paper, authors create Vehicle Node objects with a Socket attached to them to enable dynamic addition and removal of nodes in NS-3 at runtime. The proposed simulation setup is tested and validated by implementing the Green Light Optimized Speed Advisory (GLOSA) application. Based on the simulation results, authors indicate that the GLOSA offers an improvement with respect to traffic efficiency, Carbon Monoxide emissions, and fuel consumption.

The open-source project, iTETRIS, supported by the European Commission, is conducted for the large-scale assessment of Cooperative ITS applications and road traffic management solutions [142]. The paper [143] gives an overview of the simulation architecture and presents the main achievements obtained with the development of the iTETRIS platform. The 3-Blocks architecture of the iTETRIS framework [143], seen as a logical extension of the TraNS [136], proposes a real-time closed-loop coupling between SUMO and ns-3 with the use of a central control component, namely iTETRIS Control System (iCS). It is stated that by externalizing the application logic, iCS provides feasibility for users to develop their solutions without the knowledge of hidden kernel (SUMO and ns-3). To meet the requirements of the iTETRIS project, authors make extensions to both SUMO and ns-3 tools. The SUMO is extended to compute the pollutant emission or fuel consumption of the simulated vehicles. Other extensions provided for SUMO are divided into topics of intelligent rerouting, intelligent traffic lights, and advanced driver assistance systems. Concerning the networking side, a geographical addressing scheme is incorporated into the implementation of ns-3.

The V2X simulation runtime infrastructure - VSimRTI is a comprehensive and flexible framework that enables the coupling of different simulators (traffic simulator, communication simulator, environment simulator, etc.) for the assessment of various new solu-

tions or applications offered in the context of future Intelligent Transport Systems (ITS) [144]. With the easy integration of relevant simulation tools via VSimRTI, more realistic simulation environments can be created. This eventually leads us to make more reliable and proper evaluations of the developed V2X-based solutions before or during their field tests are being carried out. VSimRTI is based on the IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA), which is a standardized approach to combine different simulators with each other [145, 146]. Inspired by the HLA standard, VSimRTI makes use of the federate-ambassador concept to build up a holistic simulation setup by interconnecting various discrete event-based simulators. By this concept, each simulator is encapsulated by a federate and the ambassadors enable a bidirectional interaction between simulators and the runtime infrastructure. There exist two types of ambassadors: i) Federate ambassador provides an interface used by the VSimRTI to control simulators and convey information from other simulators. ii) VSimRTI ambassador provides an interface for the federates to access VSimRTI services, which are federation, time, and interaction management. In this architecture, the overall integrated and holistic simulation system is named as a federation of simulators (federates) [145]. The HLA standard consists of rules, a runtime infrastructure, interface descriptions and an object model template [145, 146]. The rules refer to the requirements defined for the federation and the interaction among the coupled simulators. The runtime infrastructure is the central component, which handles the communication among the participating simulators encapsulated by federates and controls the overall simulation run. In the HLA architecture, the object model templates are provided for the formal definitions of the data transferred among the federates. The integration of different simulators is enabled by the interface descriptions.

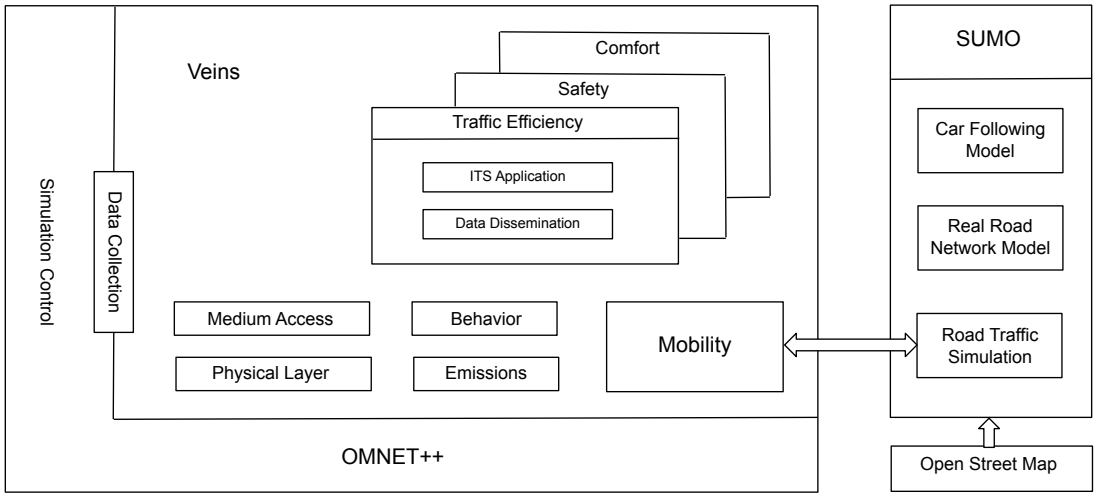

Figure 6.1: Veins Architecture [147].

6.2 Integrated and Bidirectionally Coupled Simulation Environment

6.2.1 Vehicles in Network Simulation (Veins)

Veins is an open-source, bidirectionally coupled simulation framework for vehicular network simulations, which integrates two well-known simulators from mobility and network domains: SUMO (mobility generator) and OMNET++ (event-based network simulator) [147] [148] (see Figure 6.1). While SUMO runs road traffic simulations and generates mobility traces of the vehicles; OMNET++ performs network-related simulations. The bidirectionally coupled Veins framework enables the two-way interaction between SUMO and OMNET++ during the simulation runtime. With this real-time information exchange, the realistic vehicular mobility traces generated by SUMO can be provided to OMNET++, which corresponds to the movement of network nodes. The simulation of mobile network nodes in the OMNET++ also influences the mobility of vehicles in the SUMO side through two-way interaction provided by the Veins middleware framework.

Veins framework provides a comprehensive tool suite including various Inter Vehicular Communication (IVC)-specific models to enable realistic vehicular network simulations

[147]. We summarize the functionality of the main components and models used by the Veins framework in the following:

MiXiM: Veins relies on MixiM that provides detailed models and protocols for the wireless and mobile simulations in OMNET++ [149]. Combining and extending several existing simulation frameworks, MiXiM provides modular implementations for wireless communication (wireless channel, fading, etc), mobility models, obstacle models, physical layer models, protocols at the Medium Access Control (MAC) level and localization [149]. In addition, as indicated in [149], a user-friendly graphical representation of the wireless networks and mobile nodes is enabled by the MiXiM with the support of debugging and complex scenario modeling. Readers are referred to [149] for a detailed description of the network simulation models and components provided by the MiXiM framework.

Traffic Control Interface (TraCI): The online coupling between SUMO and OMNET++ is enabled by TraCl protocol [150] [60]. TraCl provides access to the SUMO server side during the simulation runtime by using a TCP connection. During the simulation run, various mobility-related information can be retrieved from SUMO by means of the TraCl protocol, and further, the behaviors of the actors involved in the mobility simulator can be manipulated as well. It also allows us to observe the influence of vehicular network simulations on the mobility patterns of the vehicles via an online coupling.

The sequence of message exchange between SUMO and OMNET++ within the bidirectionally coupled simulation framework is depicted in Figure 6.2. As detailed in [148], the road traffic microsimulation in SUMO is advanced by the generated timesteps, and the control modules integrated with OMNET++ and SUMO buffer the commands in between the timesteps, which would enable the synchronous interaction between the two simulators. As illustrated in Figure 6.2, at each timestep OMNET++ sends buffered commands to SUMO (first phase), and then the execution of these buffered commands is triggered in the second phase. After the triggered timestep of road traffic microsimula-

tion by SUMO is executed, the resulting updated positions of the vehicles are sent back to OMNET++. With this mobility information received, OMNET++ reacts to the newly generated mobility traces by adding, removing, or moving nodes. Then, OMNET++ advances the simulation until the next timestep that allows the changing environmental conditions to affect the nodes' movements, i.e., their speed and routes. This way, the synchronized simulation of road mobility and network simulators are enabled, and the influence of inter-vehicular communication can be reflected on the instantiated vehicles' movements in a more realistic manner.

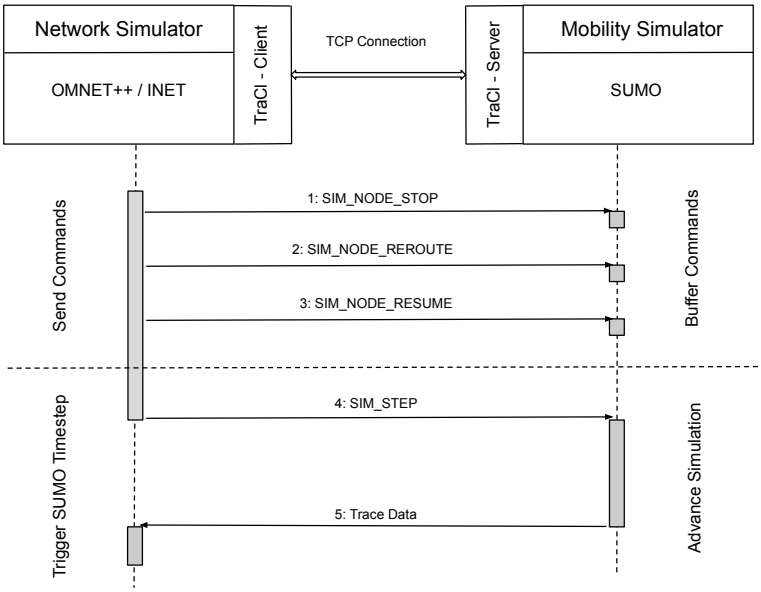

Figure 6.2: Sequence of message exchange between OMNET++ and SUMO [60] [148].

IEEE 802.11p and IEEE 1609.4 DSRC/WAVE: To realize realistic simulations for IVC-specific applications considering the unique and challenging features of vehicular networks, i.e., highly mobile topology, high velocities of vehicles, and short connection periods, Veins framework provides a protocol stack IEEE 1609 DSRC/WAVE together with DSRC PHY and MAC layer standard IEEE 802.11p [151]. The implementation of the WAVE protocol stack is provided in the Veins as free and open-source software, which provides notable improvements in terms of received data packets compared to traditional

WiFi standards (IEEE 802.11b or IEEE 802.11a), especially in the scenarios with high traffic density [151]. The implemented WAVE IEEE 802.11p model in Veins makes use of the OMNET++ network simulator and MiXiM framework as indicated in [151]. The model includes QoS channel access relying on Enhanced Distributed Channel Access (EDCA) [147]. As noted in [151], the packet error model of the implemented model is derived from [152]. The probability of a successfully transmitted packet with $18 Mbit/s$ data rate using 16-QAM OFDM is computed by the Equation 6.1 [151]. Regarding the higher layers of the implemented DSRC/WAVE stack, Veins provides models for channel hopping based on the standards, i.e., switching between the control channel (CCH) and service channels (SCH) [147]. Additionally, an application layer is provided on top of the MAC layer that enables to send WAVE Short Messages (WSMs), beacons, CAMs, and BSMs [147] [151].

$$p_{ok} = \left(1 - 1.5\ erfc\ (0.45\ \sqrt{SNIR_{min}})\right) \tag{6.1}$$

6.2.2 Simulation of Urban Mobility (SUMO)

Simulation of Urban Mobility (SUMO) is an open-source, time-discrete and microscopic traffic simulation package. It is a widely used simulation tool that enables to test various solution approaches regarding traffic management, dynamic routing, and autonomous driving. [59] [153]. SUMO allows to import real-world road networks from external sources (eg. Open Street Map) and model the traffic demand. Thereafter, city-scale simulations can be run. Being a microscopic traffic simulator, SUMO allows defining each vehicle explicitly, including vehicle's id, vehicle's type, the route that the vehicle is to follow within the road network, etc. Once the required inputs are defined and the scenario-specific configurations are done, SUMO performs large-scale simulation runs and generates various output files. The simulation results and statistics can be analyzed with the help of these output files and the necessary improvements can be made on the developed application. Undoubtedly, a very useful feature of the SUMO is that it can be coupled with other simulation tools on a common framework to simulate more complex

scenarios. For instance, it can be connected to communication simulators (OMNET++, NS3, etc.) by using the Traffic Control Interface (TraCI) framework [60], which is the case for our work as well. Throughout the mobility simulations in this study, the Krauß car-following model is used provided by SUMO.

6.2.3 Objective Modular Network Testbed in C++ (OMNET++)

OMNET++ is an open-source, extensible, modular, C++ based discrete-event network simulation framework used for modeling wired and wireless communication networks, internet protocols, queueing networks, multiprocessors, and other parallel and distributed systems [86] [154].

Modular and hierarchical network model: OMNET++'s component-based architecture enables to build large models (networks) by assembling and combining reusable components, referred to as modules [86]. As depicted in Figure 6.3, hierarchical network structures are modeled from reusable modules with unlimited nesting. The simple modules are the atomic elements of this hierarchy at the lowest level and they can not be further divided into smaller components. They can be combined to form compound modules as shown in Figure 6.3. Simple modules are the active elements of the model programmed in C++ by using the simulation library classes. They contain the algorithms and describe the behavior of the network model.

Simple modules communicate with message passing. The messages are sent and received via gates, which are input and output interfaces of the modules (depicted as small white boxes in Figure 6.3). Messages are sent either along a predefined path through gates and a set of connections, or directly to the destination module. The messages are sent through connections modeled between input and output gates of the modules. The connections can be set between the gates of two simple modules within a compound module, or between a gate of one simple module and a gate of the compound module. When a module receives a message, the local simulation time of the module advances. The connections are modeled to represent physical links by setting a set of parameters,

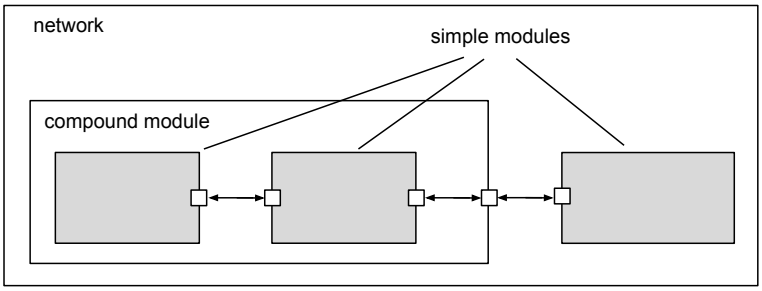

Figure 6.3: OMNET++ hierarchical network model composed of communicating simple and compound modules [86].

i.e., data rate, propagation delay, bit error rate, packet error rate. Different connection types can be defined by setting these properties accordingly, which refers to reusable channel objects.

Network Description (NED) language: The structure of the model (network) is defined by OMNET++'s topology description language, namely NED (Network Description). As detailed in the OMNET++ simulation manual [86], NED allows users to declare simple modules, and connect them to build compound modules. The compound modules can also be labeled as networks, which are self-contained simulation models. As another component type, the instances of channel objects can also be used within the compound modules. In the NED files, simple modules are declared by specifying their externally visible interfaces, that are, gates and parameters [86]. The definition of compound modules covers the declaration of their external interfaces (gates and parameters), the submodules assembled within them and the interconnection between these submodules [154]. At the highest level, the definition of a network refers to the definition of a simulation model as an instance of a module type [154]. The NED network description language supports large-scale network definitions with its features listed and described in the manual [86]: hierarchical, component-based, interfaces, inheritance, packages, inner types, and metadata annotations.

As indicated in [154], OMNET++ IDE includes a graphical editor that uses NED as its native file format and can also work with an arbitrary, even hand-written NED

code. The graphical editor is a two-way tool such that a user can edit the network topology either graphically or by using the NED source view. Indeed, it is allowed that users can switch between the graphical view and source view at any time. Rather than allowing for only fixed topologies, NED includes declarative constructs (resembling loops and conditionals in imperative languages) such that it allows for parametrized topologies [154]. With NED language, it is allowed to model common regular topologies (ring, grid, star, tree, hypercube) or random interconnection by passing numerical parameters such as size [154]. The network models defined by NED language can be mapped one-to-one to Extensible Markup Language (XML), which eases the integration of OMNET++ with other systems [154].

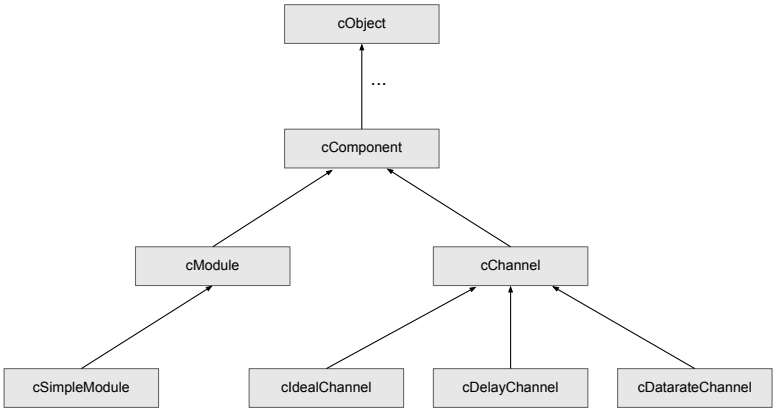

Figure 6.4: Inheritance diagram that shows the relationship of the module, channel, and component classes [86].

Components, Simple Modules and Channels: The network models in the OMNET++ simulation framework are made up from modules and connections, where modules could be simple modules or compund modules, and connections can be represented as channel objects [86]. Simple modules are the atomic and active components of the network models, and their behavior is programmed in C++ using the OMNET++ simulation class library. The channel objects, representing the connections and programmable by the user, define the channel behavior with the assigned properties for propagation and transmission time modeling, error modeling, and others [86]. The inheritance diagram

that depicts the relationship between component, module, and channel classes is provided in Figure 6.4. As seen in the diagram, modules are represented by cModule class, and channels are represented by cChannel class. Both cModule and cChannel classes are derived from the cComponent class, which is inherited from the cObject base class. Simple module types can be defined by subclassing cSimpleModule class and compound modules are defined by instantiating cModule class. As noted in the manual [86], the cModule class can be overridden with @class in the NED file, and a simple module C++ class derived from cSimpleModule can be used for compound modules. For channel behavior definition, there are three built-in channel types derived from cChannel class: cIdealChannel, cDelay Channel and cDatarateChannel as seen in Figure 6.4. New channel types are defined by subclassing cChannel or any other channel class [86].

Discrete Event Simulation (DES) in OMNET++: This section introduces a set of simulation concepts used in our implementation and describes how discrete event simulation performs in the OMNET++ framework. As detailed in [86], a discrete event system refers to a system where events (state changes) occur at discrete time instances. It takes zero time for an event to happen and it is assumed that there is nothing interesting happens between two consecutive events. In other words, there would be no state changes in the simulation model between two events occurring one after another in contrast to continuos systems, where state changes continuously take place throughout the simulation. Examples for the events considered in the computer network simulations could be the start of packet transmission, the end of packet transmission, and the expiry of a retransmission timeout [86].

The list of future events to be processed in the discrete event simulation systems are stored in data structures, called FES (Future Event Set) [86]. The working flow of discrete event simulation systems using FES is summarized in the following pseudocode (see Figure 6.5). In the initialization step, the data structures representing the simulation model are built, user-defined initialization code is called, and the initial events are inserted into FES in order to ensure that simulation can start [86]. Then, the events

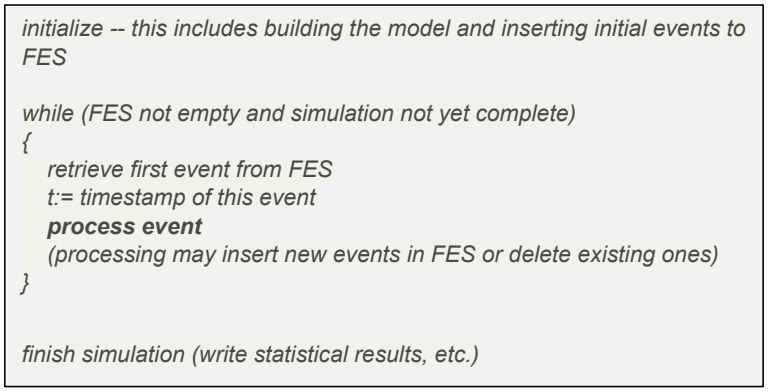

Figure 6.5: Workflow of the discrete event simulation [86].

maintained in the FES are processed in a strict timestamp such that the currently processed events do not have an impact on the previously called events. The processing step at this point refers to calling codes provided by the user. At the final step, the simulation terminates when there are no events left to be processed in the FES, model time or CPU time limit is reached, or when the statistics reach the desired accuracy as noted in the simulation manual [86]. This is the time where the user records the resulting statistics into output files.

The events in the OMNET++ are represented by using messages as instances of the cMessage class and its subclasses [86]. The messages are passed between the modules and as noted in the OMNET++ manual, the place where the event will occur is the message's destination module, and the model time when the event occurs is defined as the arrival time of the corresponding message [86]. OMNET++ employs the events maintained within the FES in arrival time order for the sake of causality [86]. To be more specific, given two messages, there are three rules defined regarding their execution: (i) the message with the earlier earlier arrival time is executed first, (ii) if the arrival times are equal, the one with the higher scheduling priority is executed first, where the scheduling priority is a user-assigned integer attribute of messages, and (iii) if the priorities are the same, the message with scheduled/sent earlier is executed first [86]. The simulation time is defined based on the SimTime class that stores the simulation time in a 64-bit integer, using decimal fixed-point representation [86]. A scale exponent global configuration

variable is used to control the resolution such that all the SimTime instances have the same resolution [86]. The exponent can take values between -18 (attosecond resolution) and 0 (seconds) each of which corresponds to a range as presented in the Table 6.1.

Exponent	Resolution	Approximate Range
-18	$10^{-18}s(1as)$	$+/-$ 9.22 s
-15	$10^{-15}s(1fs)$	$+/-$ 153.72 minutes
-12	$10^{-12}s(1ps)$	$+/-$ 106.75 days
-9	$10^{-9}s(1ns)$	$+/-$ 292.27 years
-6	$10^{-6}s(1us)$	$+/-$ 292271 years
-3	$10^{-3}s(1ms)$	$+/-$ $2.9227e8$ years
0	$1s$	$+/-$ $2.9227e11$ years

Table 6.1: Exponents with corresponding resolutions and ranges [86].

6.3 Veins Framework Extended with Multi-Objective Route Optimization

The overall simulation framework designed in this study is summarized and represented in Figure 6.6. As illustrated, the simulation concept can be broadly categorized into two parts that communicating via XML-RPC protocol: (i) the remote application server and (ii) the implemented application as part of the OMNET++/Veins framework.

The remote application server module, namely server.py, is modeled and implemented based on the Python programming language. When the server is started, the static and dynamic data layers comprising the Local Dynamic Map (LDM) are initialized and their corresponding matrices are generated based on the considered SUMO-compatible road network model. Doing so, the adjacency matrices representing the physical distance between the network nodes, as well as travel time, current congestion, projected congestion contribution, and the communication network quality are initialized. The XML-RPC server module [155] is imported into the server.py such that the communication between remote application server model and the OMNET++ application modules is established. Based on the XML-RPC framework, the functions called by the OMNET++ application

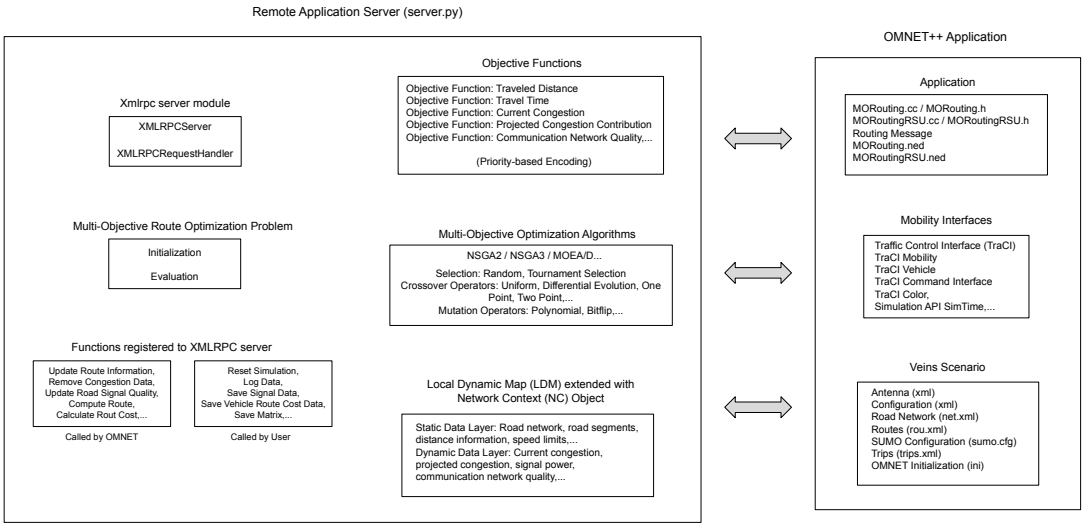

Figure 6.6: Simulation Framework.

and the functions called by the user are initialized and registered. The functions to be called by the OMNET++ refers to functions that are called by modules namely MORouting.cc and MORoutingRSU.cc. The functions contained by these modules are used to periodically report traffic information, vehicle route information, or the received signal power information by the RSUs to the server. On the other hand, the functions that are to be called by the simulation user can be used to reset the simulation, reinitialize the data layers, or save the collected simulation recordings for further analysis. Additionally, the multi-objective optimization problem is defined and initialized in the server.py module based on the Multi-Objective Optimization in Python (Pymoo) framework [87]. The number of variables (number of road network nodes), number of objectives, number of constraints, and the boundaries for the solutions are initialized. In the evaluation module of the problem model, the optimization problem objective functions and the constraints are specified. The generated solution alternatives are evaluated at every generation based on the specified objective functions and they are assigned a fitness value. The objective functions regarding traveled distance, travel time, traffic congestion, and communication network quality are defined, and the Priority-based Encoding method is used to transform the solutions into valid route representations. Lastly, the multi-objective

algorithm, i.e., NSGA-2, NSGA-3, MOEA/D, etc., is specified in the server.py module to solve the dynamic route optimization problem. The algorithm is configured by setting the parameters such as the number of generations, population size, crossover type and mutation type by using the libraries of the imported Pymoo framework.

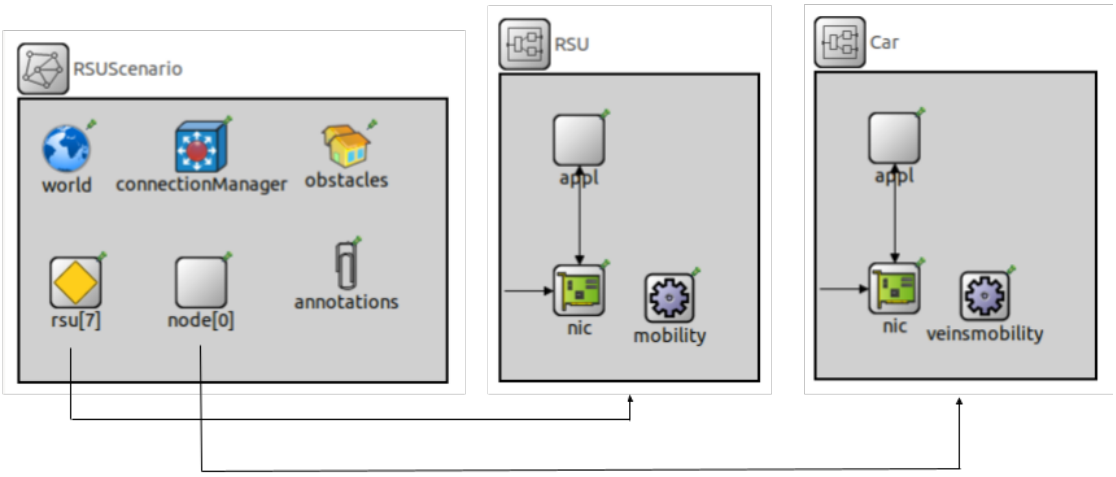

Figure 6.7: RSU Scenario.

On the other side, the OMNET++ application is designed and implemented relying on the interfaces provided by the Veins framework. Firstly, the scenario is modeled and initialized. The SUMO-compatible road network model is created, which can be generated by using the network models provided by SUMO or by converting a real road network model downloaded from the OpenStreetMap [83]. As a vehicular communication medium, we have used the IEEE 802.11p and IEEE 1609.4 DSRC / WAVE models provided by the Veins [147]. The monopole antenna model [156] is used throughout the simulations, and the analogue model is configured as Simple Pathloss Model. In the OMNET++ .ini file, the network to be simulated is specified, which we name as RSUScenario.ned network in our case (see Figure 6.7). As depicted in Figure 6.7, the network description file RSUScenario.ned comprises and refers to RSU.ned and Car.ned files. The scenario concerned in this figure has 7 RSUs. The node modules represent the vehicles such that when a vehicle object is instantiated at the SUMO side, its corresponding node module is created in the OMNET++ network simulation environment. As the vehicle

moves and changes to a new position, its movement and changed position is reflected on the node objects in the OMNET++. Mainly, the simulation parameters, RSU settings, and IEEE 802.11p parameters are specified in the OMNET++ .ini file. The simulation parameters section includes the size of the background area, simulation time limit, etc. The RSU settings section specifies the locations of the RSU instances within the borders of the background area. The parameters regarding the IEEE 802.11p model include the maximum transmission power (mW), bit rate (Mbps), minimum signal power (dBm) and antenna properties. In addition to the configurations regarding the vehicular communication network, the SUMO-compatible road network (.net.xml), and the initially generated trips (.trips.trips.xml) and routes (.rou.xml) of the vehicles are also included in the Veins scenario directory. For instance, to generate the background traffic, first the trips are randomly generated between a specified simulation time interval using the randomTrips.py component of the SUMO [59]. Then, based on the created trips file, the initial routes of the vehicles are determined using the duarouter component, which relies on a shortest-path route computation method [59]. In the end, the routes of each vehicle are stored in the (.rou.xml) file together with the departure time information. The application in the OMNET++ side is mainly implemented in the MORouting.cc and MORoutingRSU.cc modules based on c++ programming language. The MORouting.cc module defines the behavior of each vehicle object instantiated within the Veins simulation environment. The main function of the MORouting.cc module, namely handlePositionUpdate, is called and run whenever a vehicle in the scenario moves by one simulation unit. Hence, the functions defined within the MORouting.cc class are executed for any vehicle move. In this module, whenever a routable vehicle (a vehicle that is periodically rerouted by a multi-objective route optimization methodology) is instantiated, an initial route request is sent for this vehicle to the server. Then, the calculated route is assigned to this vehicle at the very beginning of its trip, and the data layers at the server are updated accordingly based on the newly calculated route information. For instance, the current congestion and the projected congestion contribution data layers are updated based on the received route information at the server.py. After this initial-

ization process, new routing requests are periodically sent for each one of these routable vehicles throughout the simulation. The route information of the routable vehicles is continuously sent to the server in predefined intervals and also whenever they change to a new road segment (edge). The initialization process and the periodic route calculation process are not applied for the unroutable vehicle types (vehicles that are not rerouted throughout the simulation). However, the current route information of these unroutable vehicles comprising the background traffic is also sent to the server whenever they change to a new road segment (edge). In this way, the data layers regarding traffic congestion are frequently updated by every vehicle in the scenario. Additionally, every vehicle in the scenario periodically broadcasts its current road information to the RSUs and other vehicle nodes via the WAVE Short Message (WSM) protocol provided by Veins. In the MORoutingRSU.cc application module, whenever a message is successfully received by an RSU, the message content is extracted and the route information received by the vehicle is reported to the server, which is then used to update the dynamic data layers. The Received Signal Strength (RSS) information, or signal power information, is also extracted from the arriving message and sent to the server by the RSU. By this way, the data layer, maintaining the signal power and communication network cost information, is updated. Throughout the simulation, until the last vehicle instance leaves the simulation, the multi-objective route planning mechanism is periodically run for the routable vehicles based on the most up-to-date local dynamic map data structure.

7 Analysis and Evaluation

This chapter presents the simulation test results we have obtained for different instances of the multi-objective route optimization problem by considering road network models provided by the SUMO mobility simulator and a real-world road network scenario provided by the OpenStreetMap [83]. The simulations are performed by varying the set of objectives, constraints, and road network scenarios. We provide the analysis and evaluation of the simulation outcomes for these different case studies.

7.1 Road Network Models

In the simulation studies, we have considered two types of road network scenarios, one of which is generated by means of the abstract network generation method of SUMO and the other one represents a region of Berlin. As an abstract road network model, we have generated a spider-like network by using the netgenerate component of SUMO [59]. The netgenerate is a tool of SUMO that is used to generate three types of abstract networks, namely grid-like network, spider-like network and random network [59]. The spider-like network model is used for this study. An example command for generating a spider-like road network is given in the Listing 7.1 below. According to this command, a spider-like network is generated with 10 arms, 6 circles, and a distance of 100 meters in between the circles. Figure 7.1 shows the generated road network with 37 nodes and 144 edges, which is viewed by using the graphical editor of the SUMO namely netedit [59].

List of Listings 7.1: The command to generate spider-like road network in SUMO.

```
$ netgenerate --spider --spider.arm-number=10 --spider.circle-number=6 --
    spider.space-radius=100 --output-file=spiderNet.net.xml
```

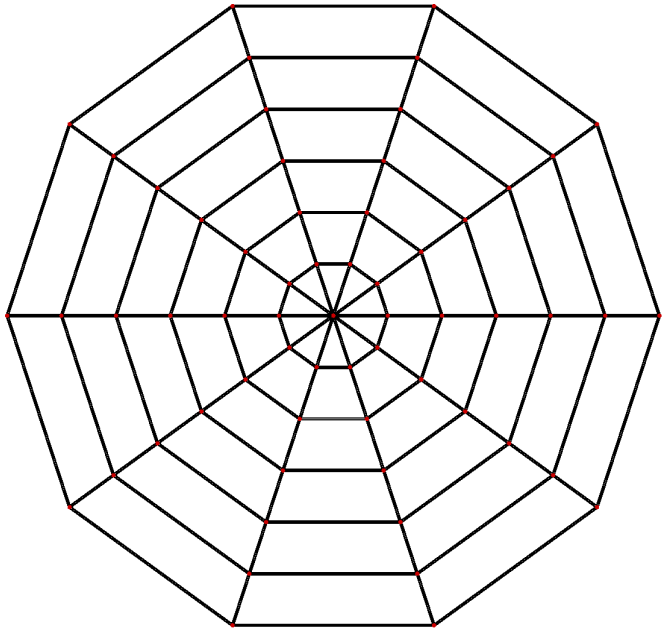

Figure 7.1: Spider-like abstract road network model viewed by the netedit tool of SUMO.

Additionally, we have used a real-world road network model representing a region of the Berlin city center. Figure 7.2 shows the corresponding real-world road network model with 248 nodes and 507 edges, which is viewed by the sumo-gui tool. The road network data is downloaded from the well-known and widely used map data source OpenStreetMap [83]. Firstly, the map data corresponding to the selected region is downloaded from the OpenStreetMap in .osm format, i.e., as Berlin.osm file. Then, the OpenStreetMap .osm file format is converted to SUMO network format (.net.xml) by using the netconvert tool of the SUMO. More specifically, the Berlin.osm file is converted to Berlin.net.xml file by using the command given in the Listing 7.2 below. In this way, a suitable network file format is generated that can be imported to SUMO. The command provided in the Listing 7.2 is a general call for generating SUMO-network from .osm data, but there are various additional options of netconvert tool listed in the SUMO

website [59]. By setting these options accordingly, users are able to provide additional data layers to the SUMO-network file. In this work, we have filtered out the irrelevant data such as railroads and removed the redundant nodes in the road network model.

List of Listings 7.2: The command to generate SUMO network file from OpenStreetMap file format.

```
$ netconvert --osm-files berlin.osm.xml -o Berlin.net.xml
```

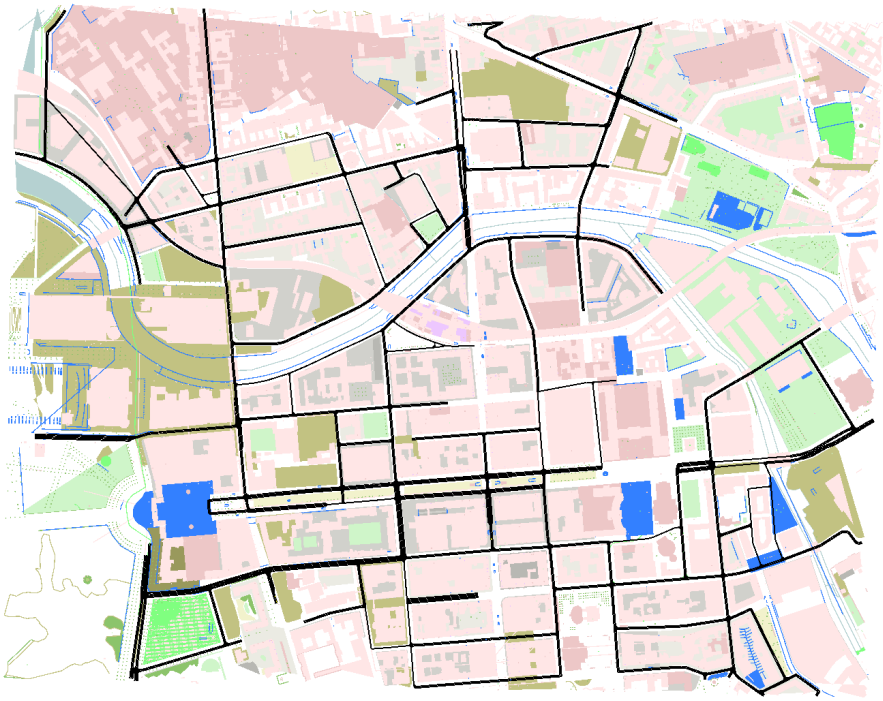

Figure 7.2: Real-world road network model representing a region of Berlin city center, viewed by the sumo-gui tool.

7.2 Simulation Tests by using Spider-like Road Network Model

Case study 1: In this case study, we have considered a small-size road network with 31 nodes and 96 edges as seen in Figure 7.3. This road network model is formed by removing out the nodes and edges of a spider-type network. The reason for using a small-size network at the beginning of our simulation studies was to quickly observe if

the multi-objective route optimization algorithm can find optimal routes by considering the dynamically changing congestion parameter. For this purpose, a heavy traffic is generated along the straight road that connects the leftmost edge and the rightmost edge of the road network. The vehicles without rerouting capability are expected to follow the straight road without avoiding congestion, and the vehicles with rerouting capability are expected to turn on right or left sideways to avoid congestion. To ensure that enough amount of congestion occurs along the straight road, additionally, we have placed traffic lights at four junctions as can be seen in the Figure 7.3.

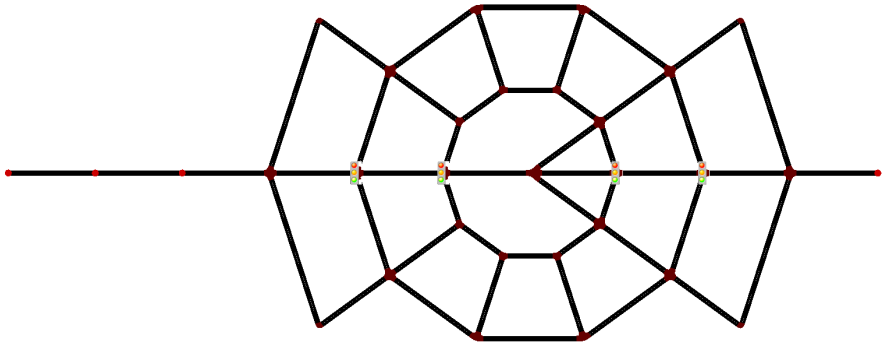

Figure 7.3: A basic road network model created by filtering out a spider-type network model.

In this road network scenario, 15 vehicle instances are created in total, all of which are traveling from the left-most edge to the right-most edge of the road network. The first 10 vehicles depart at a time in between $0 - 20sec$ period, and the last 5 control vehicles depart at time instances $t = 50, 51, 56, 58, 60sec$. The first 11 vehicles are set to be not routable, and the last 4 vehicles are set to be routable. This way, the route costs of two vehicles (one vehicle without rerouting capability that departs at $t = 50sec$, and one vehicle with rerouting capability that departs at $t = 51sec$) are compared. The route congestion cost values for these two vehicles are illustrated in Figure 7.4 (a). The congestion cost of the currently followed routes of the two vehicles is recorded as the vehicles change to a new edge. As seen in Figure 7.4 (a), the route congestion cost of the routable vehicle is always lower than the route cost of the vehicle without dynamic routing capability. In this case study, the multi-objective optimization problem

instance for the routable vehicles is defined with two objectives and two constraints. The objectives are to minimize the total traveled route distance and the current congestion cost of the traveled route. The first constraint is that the route congestion cost should be lower than or equal to 0.2; the second constraint is that distance of the offered route solution needs to be lower than or equal to 1.5 times the shortest-path distance from the current location of the vehicle to the destination location. As seen in Figure 7.4 (b), the ratio between the offered route distance and the shortest-path route distance is always found to be lower than 1.5 for all the four routable vehicles. To dynamically compute the shortest-path distance from the current location to the destination location of a vehicle, we have used Dijkstra's algorithm, presented in the article [157]. In this scenario, the NSGA-2 algorithm is modeled and used for the dynamic route optimization with population size of $N = 200$, maximum generation number of $N_g = 50$, crossover probability of $p_c = 0.9$ and mutation probability of $p_m = 0.05$.

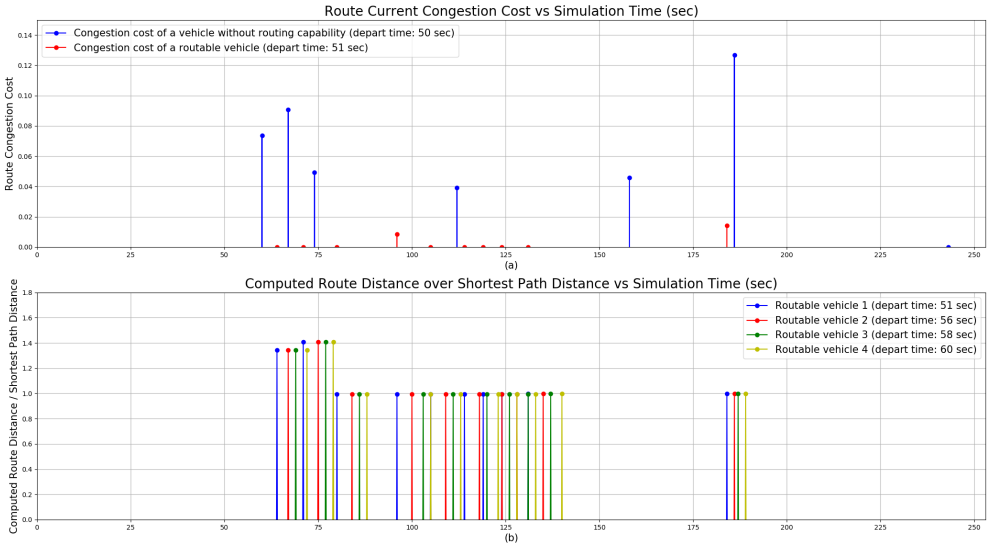

Figure 7.4: (a) Two vehicles' route congestion cost. (b) The ratio of computed route distance over shortest-path route distance for the 4 routable vehicles.

In the second part of this case study, the last 5 control vehicles are observed. First,

simulation is run with all the 15 vehicles, including the last 5 control vehicles, without rerouting capability. Then, only 5 vehicles out of 15 vehicles (control vehicles) are periodically rerouted with a period of $30sec$. The vehicles without rerouting capability means that they are following a shortest-path algorithm without considering dynamic metrics to reach their destination. On the other hand, the rerouted vehicles periodically use a multi-objective route planning algorithm by considering both distance and congestion parameters. As can be seen in Figure 7.5, the vehicles (yellow ones) without rerouting capability follow the straight road, on the other hand, the vehicles with rerouting capability (green vehicles) avoid the congested road (see Figure 7.6).

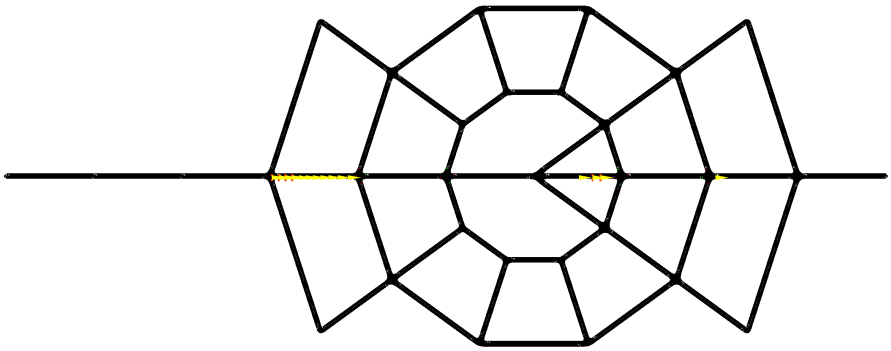

Figure 7.5: The simulation status recorded at $t = 93.00sec$. All 15 vehicles in the scenario are set to be without rerouting capability.

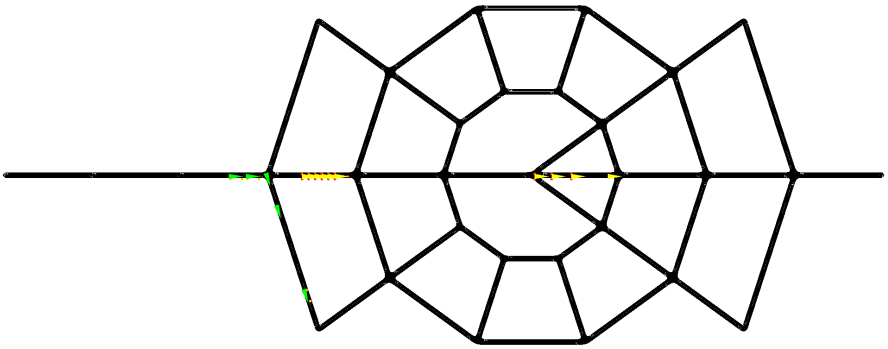

Figure 7.6: The simulation status recorded at $t = 83.900sec$. The 5 of the 15 vehicles (green vehicles) have the rerouting capability.

To illustrate this, Figure 7.7 shows the travel time, waiting time and time loss values of the control vehicles obtained as a result of the simulations. The travel time parameter

corresponds to the total time spent to travel the route, the waiting time is the time that vehicle's speed is below or equal to $0.1m/s$, and time loss parameter is the time lost because the vehicle travels below its ideal speed [59]. While calculating the waiting time and time loss parameters, the scheduled stops are not taken into account [59]. In SUMO, the desired speed of the vehicle instances are defined by using the speed factor or speed deviation attributes [59]. As indicated in the SUMO User Documentation [59], if a vehicle's speed factor is set to be 1.2, then this vehicle can drive up to 20% above the speed limit (legal speed); or if a vehicle's speed factor is 0.8 than this vehicle is to travel with a speed that is always 20% below the speed limit. In this scenario, the passenger vehicles' speed deviation attribute is taken as 0.1, which is the default value.

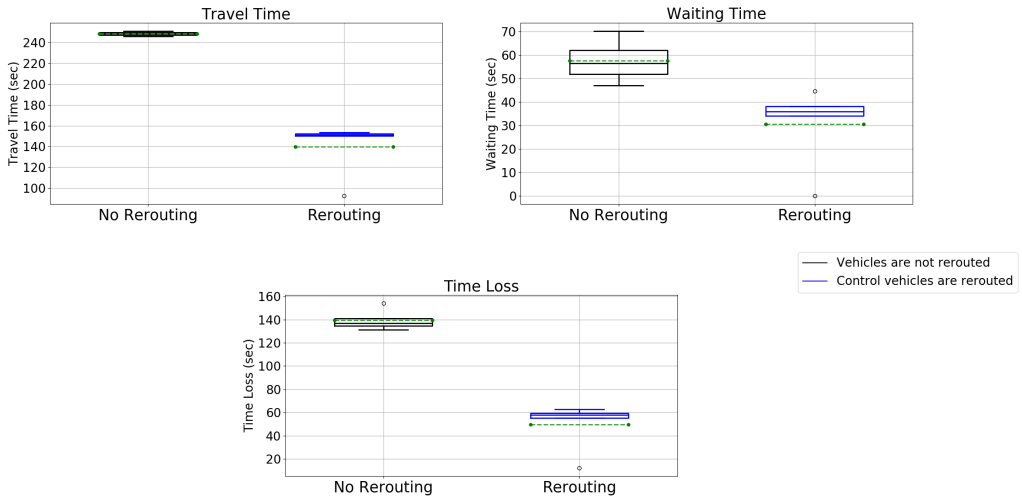

Figure 7.7: The travel time, waiting time, and time loss values of 5 control vehicles involved in the scenario.

In Figure 7.7, it is seen that when there are reroutable vehicles (vehicles with green color) involved in the simulation, their travel time, waiting time, and time loss values decrease. The reason for this is that when vehicles have been rerouted by considering the congestion metric, they have avoided the congested straight road in the middle of the road network and reached their destination in less amount of time. Throughout the simulation, the current congestion values of the routes followed by the control vehicles are

recorded as they change to a different road segment (edge of the road network graph). These current congestion values are recorded when the 5 control vehicles are all not routable, and when they are set to be routable. The mean current congestion values of the routes followed by the control vehicles are calculated for the two cases and illustrated in Figure 7.8. As seen in Figure 7.8, the mean current congestion values of the routes of the control vehicles decrease when they are reroutable and pass through road segments with lower congestion cost metric.

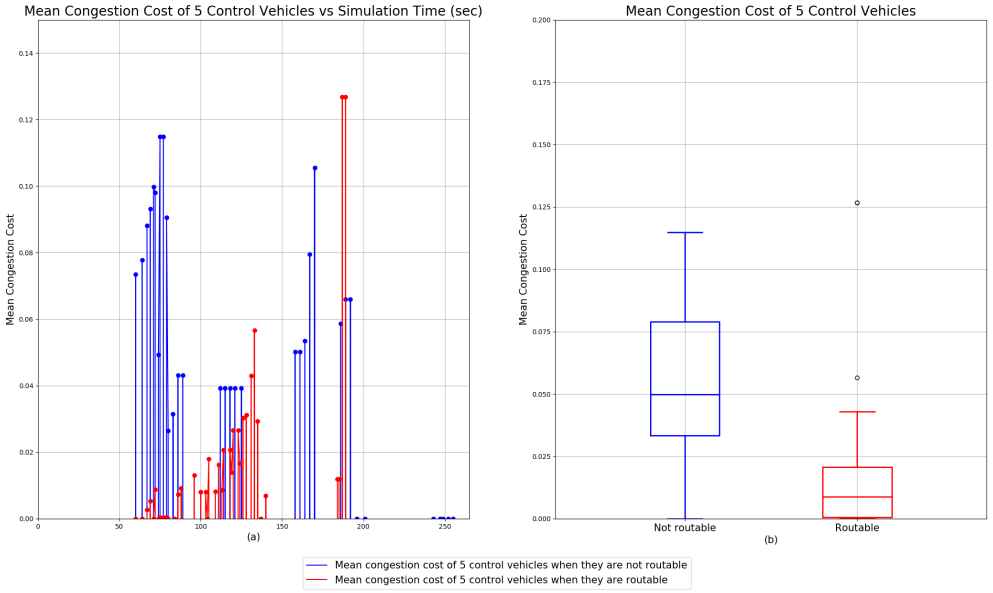

Figure 7.8: Mean congestion cost of the 5 control vehicles when they are not routable and when they are set to be routable.

Case study 2: In this case study, we have considered a spider-like road network model with 61 nodes and 240 edges as seen in Figure 7.10. We have created 500 trips as background traffic (represented by the yellow vehicles), which are departed at a time in between $0 - 500sec$ period. The start and end locations of these trips are determined randomly. Then we have defined 30 additional trips all of which have been departed from the left-most edge towards the right-most edge of the straight road in the middle of the road network. We have placed 7 RSUs at the bottom part of the road network as

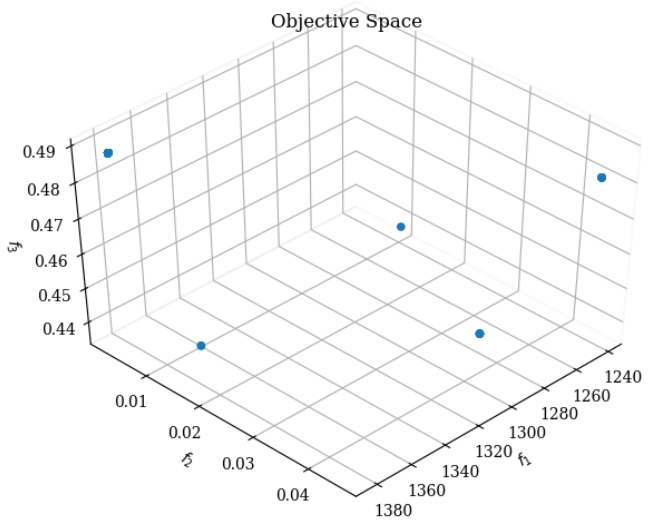

Figure 7.9: Illustration of the optimal solutions in the objective space found by the NSGA-2 as a result of a route request sent by a routable vehicle instance.

seen in Figure 7.10. The first 15 trips (represented by the red vehicles) of the additional 30 trips depart at a time between $250 - 265sec$, and the other 15 trips (represented by the green vehicles) depart at a time between $265 - 280sec$. All the vehicles comprising the background traffic (yellow vehicles) and the first group of 15 additional trips (red vehicles) are not rerouted. The second group of 15 additional trips (green vehicles) is rerouted by the network-aware multi-objective route optimization algorithm. In this scenario, we have considered three objectives which are set to be minimizing traveled distance, current congestion level, and communication network-related cost of the route. The constraints are defined regarding the total traveled distance and the overall route network quality. According to these constraints, the distance of the offered route should be lower or equal to 1.7 times the shortest-path distance from the current location of a vehicle to its target location, and the overall network cost of the offered route needs to be lower or equal to 0.5 (and 0.4 to test another case). The NSGA-2 parameters are set to be the same as the previous case study, as we have obtained sufficiently good results.

As illustrated in Figure 7.9, the adapted NSGA-2 algorithm finds out the most optimum solution set based on the set of defined objectives and constraints.

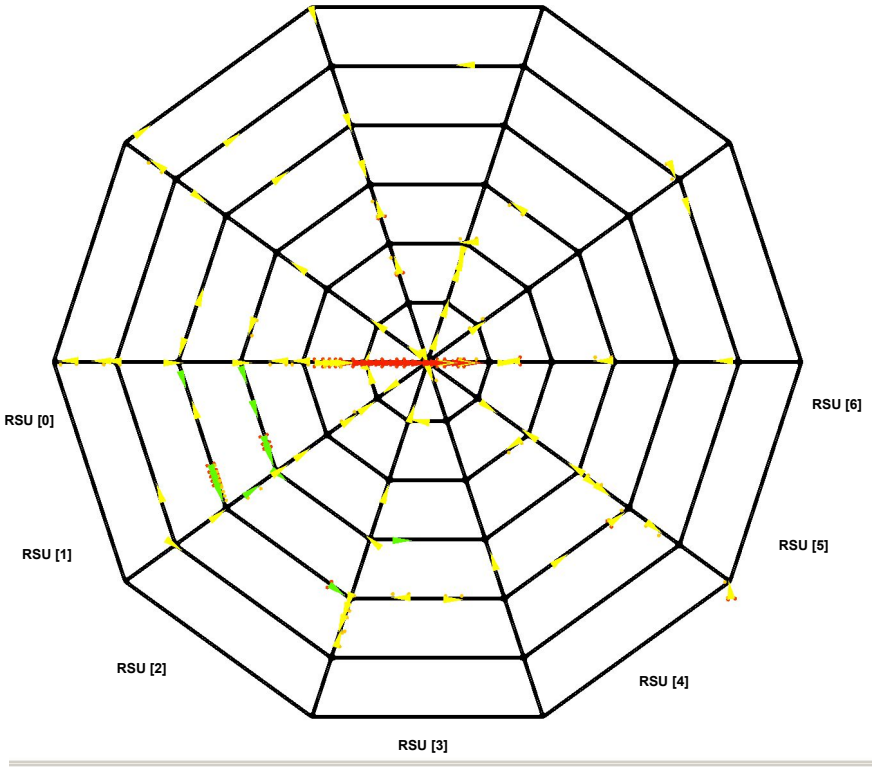

Figure 7.10: Spider-like road network with RSUs deployed at the lower part.

In this case study, vehicle nodes periodically broadcast their road information, which arrives at RSUs and other vehicles around, as seen in Figure 7.11. The RSUs receive messages with higher signal power (dBm) from the vehicles that are traveling on the edges closer to RSU locations. In other words, the signal power values assigned to the edges at the lower portion of the road network are higher, compared to the ones at the upper portion. Accordingly, as an intuitive observation, it is seen that vehicles with rerouting capability (green vehicles) turn towards the RSUs and travel to their destination through the lower part of the road network. More specifically, they follow the routes with higher signal power and lower communication network cost. To quantitatively support this intiutive observation, the signal power (dBm) and network cost values of the routes traveled by the vehicles are recorded throughout the entire simulation time. When

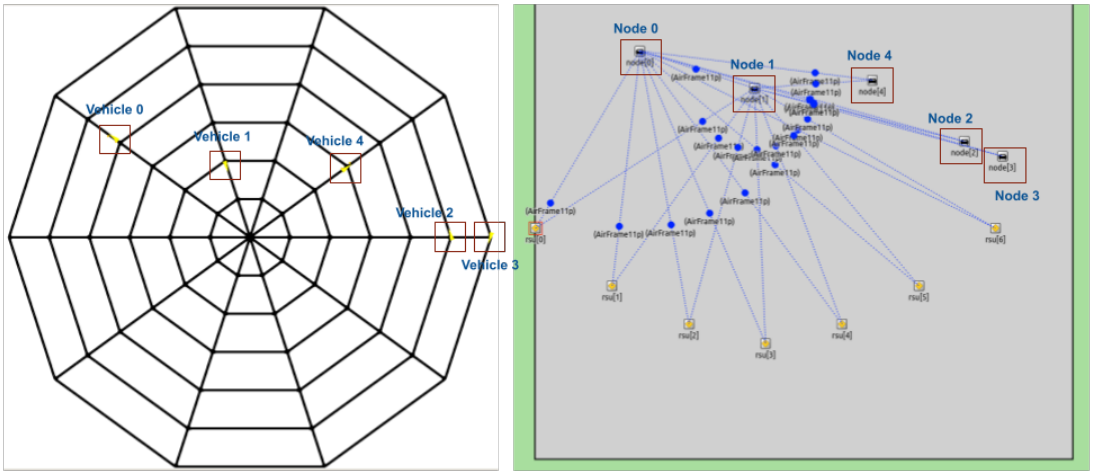

Figure 7.11: Illustration of the broadcast of road information by the vehicle nodes in the spider-type road network scenario. There are 5 vehicle instances and their network node representations at the very beginning of the scenario.

the vehicles change to a new road segment (edge), the total signal power and network cost value of the routes that they are currently following are recorded. Figure 7.12, comparing the simulation recordings of two vehicles (one without rerouting capability and one with the rerouting capability) departed almost at the same time, illustrates that rerouted vehicle's routes have higher signal power (dBm) and lower network cost. Likewise, Figure 7.13 compares the mean signal power and mean network cost resulting from the routes traveled by two groups of vehicles: (i) the first group includes 15 vehicles without rerouting capability (red vehicles on Figure 7.10) and (ii) second group is the 15 reroutable vehicles (green vehicles on Figure 7.10). The route cost measurements are collected using the same methodology such that when vehicles change to a new road segment (edge) their current route information is recorded. Similarly, as illustrated in Figure 7.13, the mean signal power is found to be higher and the mean network cost is found to be lower for the reroutable vehicles.

We have modeled two scenarios such that in the first scenario the distance-related constraint is included in the problem definition, and the communication network-related constraint factor is taken as 0.5. For the second scenario, distance-related constraint is not involved and the network-related constraint factor is set to be 0.4. In both Figure 7.12

and Figure 7.13, it is seen that when the distance constraint for the reroutable vehicles is not included in the problem definition and the network-related constraint factor is taken as 0.4, the difference between the results of regular vehicles and the reroutable vehicles becomes more apparent. When there is no distance constraint, the reroutable vehicles (green vehicles) are able to travel through more distant roads and pass through closer to RSUs. By this way, they travel through road segments with higher signal power and lower network cost assigned, compared to the case where there is distance constraint. The simulation results regarding mean signal power and mean network cost of these two groups of vehicles are also presented in Figure 7.14 considering both scenarios.

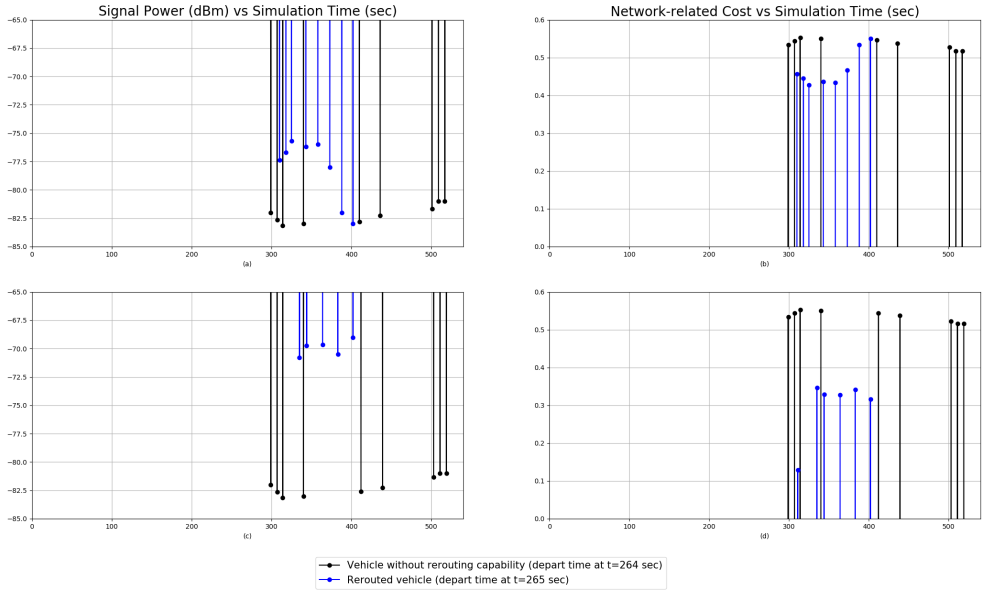

Figure 7.12: Signal power and network cost when there is distance constraint and when the network constraint factor is 0.5 ((a), (b)). Signal power and network cost when there is no distance constraint and when the network constraint factor is 0.4 ((c), (d)).

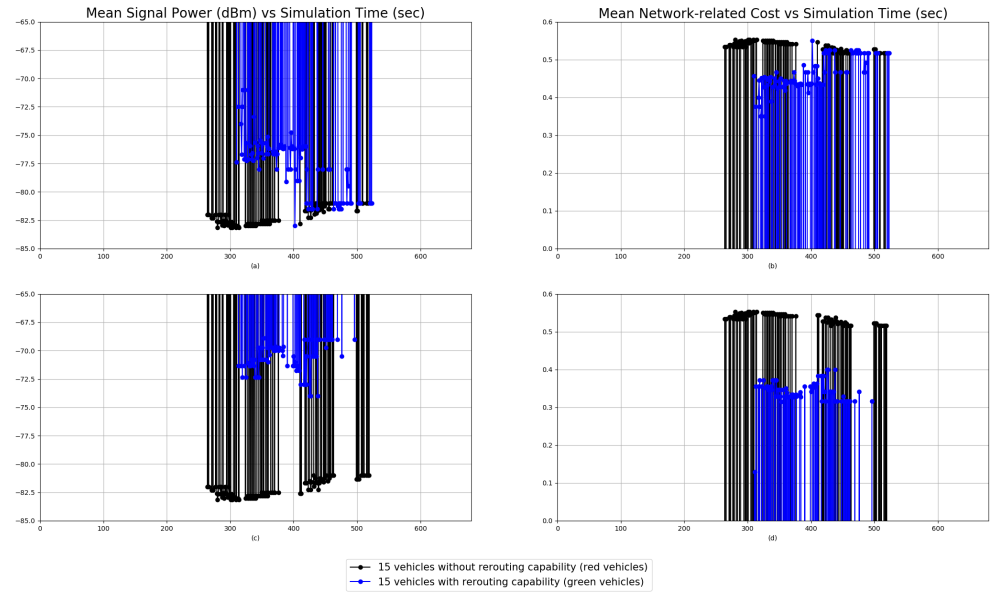

Figure 7.13: Signal power and network cost when there is distance constraint and when the network constraint factor is 0.5 ((a), (b)). Signal power and network cost when there is no distance constraint and when the network constraint factor is 0.4 ((c), (d)).

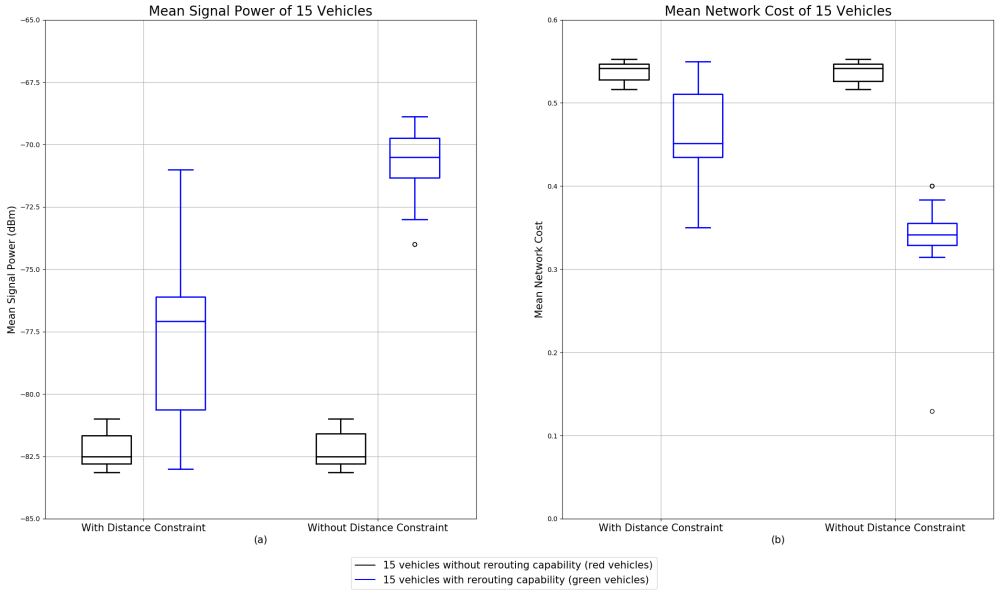

Figure 7.14: (a) Mean signal power. (b) Mean network cost.

7.3 Simulation Tests by using Real-World Road Network Model

In this case study, we have performed simulations by considering a real road network with 248 nodes and 507 edges. The road network represents a region in the Berlin city center as can be seen in Figure 7.2. We have randomly generated 500 trips for a $1000sec$ period and observed the routes of 10 vehicles among all the trips. Five of these vehicles are departed at a time in between $480 - 500sec$ without rerouting capability. The other group of five reroutable vehicles is departed at a time in between $500 - 540sec$. The start location, destination location, and the initial route of these 10 vehicles are configured to be the same at the beginning of the simulation. We have considered three objectives and two constraints for the reroutable vehicles. The objectives are set to be minimizing total traveled distance, current congestion cost, and the projected congestion cost. The constraints are defined regarding the distance and current congestion cost metrics. The maximum ratio of the calculated route distance over the shortest-path distance from the vehicles' current location to their destination location is set to be 1.5. In addition, the maximum current congestion cost value on any edge of the calculated route is set to be 0.25. Same as the previous cases, the NSGA-2 algorithm is used for this scenario. The number of generations, population size, crossover probability, and mutation probability are set to be 50, 200, 0.9, and 0.05 respectively. During the simulation run, it is observed that vehicles rerouted by using the multi-objective evolutionary algorithm avoided the congested route regions and arrived at their destination at a shorter time period compared to vehicles without rerouting capability. The rerouted vehicles followed a longer-distance path as expected. The simulation outcomes of this case study are summarized in Figure 7.15 and Figure 7.16. In Figure 7.15, it is seen that travel time, waiting time, and time loss values are found to be lower for the rerouted vehicles. Throughout the simulation, the distance and congestion-related route cost values of all the vehicles are recorded at the time when vehicles change to another road segment. As shown in Figure 7.16, the current congestion cost, projected congestion cost, and average projected congestion cost of the rerouted vehicles remains lower than vehicles without rerouting capability most of the time. Further, the calculated route distance of the

rerouted vehicles is mostly not higher than 1.5 times the shortest-path distance at any time of the simulation. There are two exception measurements as seen in Figure 7.16 (d) such that two of the recorded ratios are found to be higher than 1.5. Most likely, these exceptions occurred when the optimization algorithm could not find solutions satisfying the given constraints, and in this case, the corresponding vehicle continued to follow the previously calculated route.

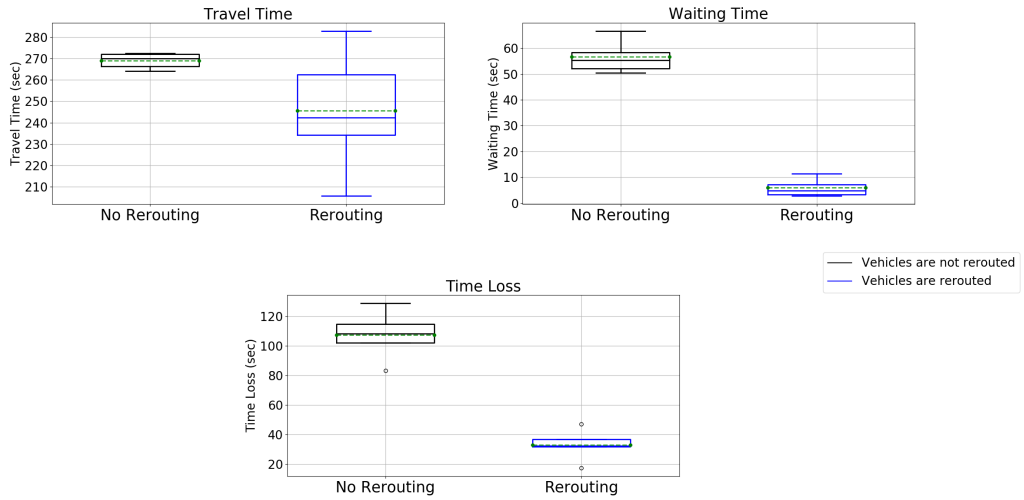

Figure 7.15: The travel time, waiting time, and time loss values of 5 vehicles with and without rerouting capability.

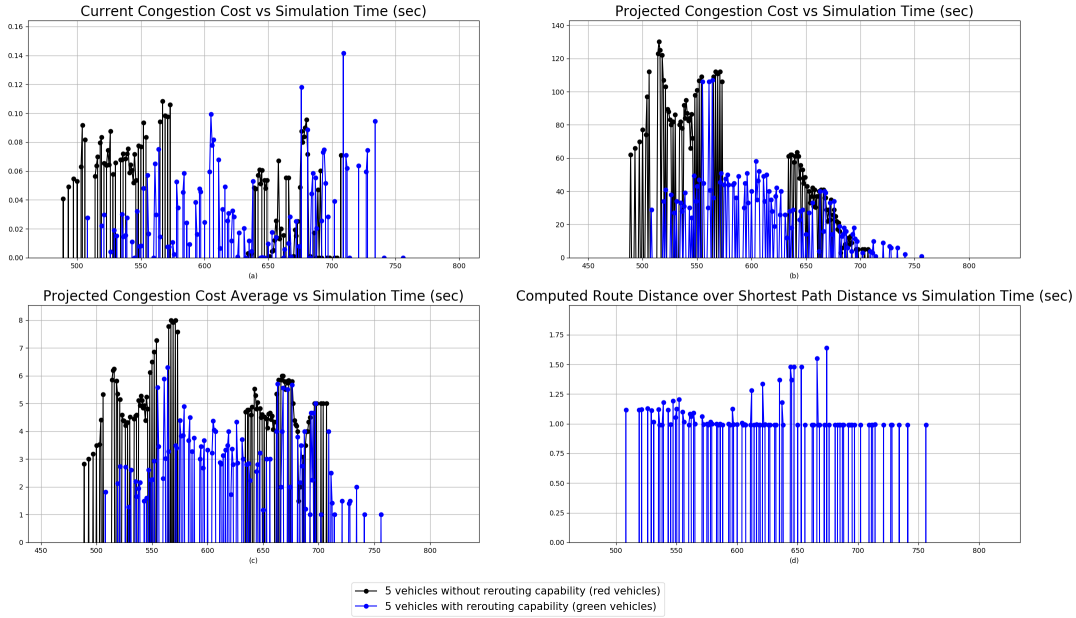

Figure 7.16: (a) Current congestion cost, (b) projected congestion cost, (c) average projected congestion cost of the routes followed by the routable and unroutable vehicles throughout the simulation. (d) The ratio of calculated route distance over shortest-path distance for the routable vehicles.

8 Conclusion

This research study argues that route planning techniques and road traffic management schemes might need to be revisited by considering the requirements of autonomous and connected driving. At this point, we believe that seamless connectivity both within and around the vehicle is one of the key enablers of the advanced autonomous driving applications, i.e., vehicle platooning, advanced driving, extended sensors, and remote driving, which has to be concerned as an additional metric in the next-generation route planning architectures [2]. The planned routes of the autonomous and connected vehicles have to be dynamically adjusted in a way that vehicles have efficient and reliable connectivity with a certain level of QoS throughout their travel until arriving at the destination location [2]. Based on this idea, we design, implement and propose a multi-objective route planning framework, which enables us to incorporate both traffic-related and communication network-related measures into the route optimization problem model.

Firstly, a bidirectionally-coupled simulation framework is set up, which includes a mobility simulator for the simulation of vehicular traffic and a communication network simulator for the simulation of wireless data transfer within the vehicular road environment. For the simulation of vehicular traffic, we have used the well-known mobility simulator, namely Simulation of Urban Mobility (SUMO). To simulate the communication network simulations we have used the commonly used network simulator of Objective Modular Network Testbed in C++ (OMNET++). These two simulators are integrated by means of the Vehicles in Network Simulations (Veins) framework. In doing so, two-way interaction between the mobility simulator and the network simulator is settled. An urban scenario is modeled by using these two simulators, including autonomous and

connected vehicles periodically rerouted throughout the road network, roadside units that communicate with the vehicles and the remote application server. This integrated simulation framework enabled to dynamically sense the traffic and communication network conditions of the roads. The measured traffic condition information and network status information are sent to the remote application server, resembling the Cooperative Intelligent Transportation Systems (C-ITS) deployed in the urban areas. Additonally, a continuously updated data structure, representing a local dynamic map (LDM), is built at the server-side. This multi-layer data structure keeps different static and dynamic information regarding the road traffic conditions and availability of wireless access technologies.

As an extension to these tools, we have established a modular framework that allows us to incorporate and remove different objectives or constraints to/from the route planning optimization problem model. For instance, by using this modular structure, one can easily model objective functions or define constraints regarding the relevant metrics of the vehicular route planning problem, i.e., traveled distance, travel time, current congestion level and projected congestion contribution level, as described in the previous chapters of this study. By considering the connectivity requirements of autonomous vehicles, communication network quality metric can be incorporated into the problem instance. In the modeled urban scenario, every vehicle instance periodically broadcasts its current road information to the closest RSU. Upon successfully receiving the message from a vehicle, the corresponding RSU node extracts a signal strength value (or signal power in dBM) out of the message, which quantifies the network quality level at the road where the message is sent. The RSU then sends the received signal strength measurement to the application server together with the corresponding road information, and in this way, the network layer of the dynamic data structure maintained in the server is updated. Throughout the simulation, the same procedure is repeated by every RSU in the scenario whenever they successfully receive road information from the vehicles.

When it comes to congestion metrics, we have concerned both current congestion level and projected congestion contribution level on the roads in this work. The (near) real-

time traffic information collected by the vehicles is reported to the application server both periodically and when the vehicles change to a new road segment. Using this information, the current congestion level on each road segment is estimated and continuously updated. At the time that an alternative route is calculated by the server and sent to a vehicle, the projected congestion contribution values along the new route of the corresponding vehicle are updated in the relevant data layer of the server. The highest congestion contribution value is assigned to the next road segment that the vehicle will be visiting after receiving the alternative route. The following roads up to the last road segment are assigned with lower values in decreasing order, as they will be visited at later times by the vehicle. Further, the projected congestion contribution values assigned to the edges involved in the new alternative route are updated with new values as the vehicle moves on its route and changes to another edge. The same procedure is dynamically applied for every vehicle involved in the scenario such that an aggregated projected congestion contribution value is calculated for every edge resulting from the set of vehicles that are to be passed along the corresponding edge. To realize this, we have determined and recorded the registered vehicles for every edge at certain periods of time, where registered vehicles refers to a set of vehicles that have the corresponding edge ID in their currently followed route. The aggregated projected congestion contribution value of an edge is calculated by using the route information of every registered vehicle to this edge at a time. As was previously highlighted, by means of traffic congestion estimation for the future, it is aimed to realize network-wide, global traffic optimization, rather than rerouting the vehicles based on individually optimized routes in a greedy way.

With the traffic condition and communication network status information received, a cost term is determined for every road segment associated to all the dynamically changing metrics mentioned above. A cost matrix is created specific to each metric and continuously updated based on the collected information. In addition to the cost matrices standing for the dynamic metrics, we have created the distance cost matrix of the road network as well. Using these cost matrices, the most optimum set of route solutions are dynamically calculated for the vehicles by using the Multi-Objective Evo-

lutionary Algorithm (MOEA) approach. For instance, when a route request is received from a vehicle, the optimum routes with minimized total distance, current congestion, aggregated projected congestion contribution, and communication network-related cost can be calculated by the adapted MOEAs. By using the provided modular planning framework, different types of algorithms in this context can be adapted with the required configurations, i.e., Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D), Non-dominated Sorting Genetic Algorithm 2 (NSGA-2), and Non-dominated Sorting Genetic Algorithm 3 (NSGA-3). For this thesis report, we have mainly used the NSGA-2 algorithm as it provided promising performance with the modeled road network scenarios. The group of vehicles subscribed to the designed route planning scheme is periodically rerouted by using these state-of-the-art optimization techniques for different cases, with a different set of objectives and constraints. The algorithms and framework are validated on both synthetic and real-world road network models.

8.1 Future Research Directions

A valuable extension to this work could be integrating the planned routes of vehicles into the network resource management entity of communication service providers for proactive resource allocation. As depicted in Figure 1.1, the feedback loop between the network-aware route planning component of autonomous vehicles and the SDN-based resource controller of the network provider can be implemented for this purpose. It should be highlighted that autonomous and connected vehicles have the advantage of having their route programmatically determined and adjusted, unlike human-operated vehicles. Using this benefit, the planned and deterministic trajectories of the autonomous vehicles can be provided as input to the network resource management entities such that SDN-based controllers can preemptively allocate network resources (i.e., available bandwidth, flow entries) along the trajectories of the vehicles. Thus, the trajectory-based proactive network resource management can be implemented on the network side as a future work for this study [2].

As another research dimension, the presented route planning mechanism can be improved to be a scale-invariant scheme. When there are larger-scale road network scenarios, the road network can be divided into multiple sub-regions, and there could be local server node(s) specific to each region. These locally distributed server nodes would be responsible for the route computations only within their area of concern. When the sub-routes calculated in each region are combined, the resulting route would be the overall planned route of the vehicle from its start location to its destination location. This way, the computation load on the central remote application server can be offloaded to local servers. Additionally, the communication load can be reduced when compared to a scenario where there is only one central server. In this case, there could be a threshold between reducing the computation power and computing the globally optimum route, which requires further analysis.

The ridesharing concept would provide further benefits when it is integrated into large-scale AMoD systems, i.e., reduced passenger waiting time, congestion, energy consumption, and traffic-related pollution [56] [158]. In an AMoD system where ridesharing is enabled, multiple commuters traveling in the same direction are matched and assigned to the same available vehicle. In this way, the size of the AMoD fleets can be reduced and fewer vehicles can serve a similar amount of demand. As indicated in [158], two objectives can be considered for the design of AMoD system with ridesharing, which are maximizing quality of service and minimizing the operation cost. From the users' perspective, the driving comfort throughout the ride is to be maximized, i.e., the total travel time and waiting time of the passengers need to be minimized. When the operation of the AMoD system is concerned, the fleet size and total energy consumption need to be minimized [158]. Hence, the route planning mechanism of automated and connected vehicles can be extended to consider the operation of a larger scale and more realistic AMoD system model with ridesharing included.

Finally, the motion planning of autonomous and connected vehicles can be considered as a whole by covering all the planning layers in the hierarchical decision-making mechanism (see Section 2.2.1) as future work. Recall that this study focuses on the higher-level

route planning of vehicles from their current location to their final destination location, and aims to find out the most efficient sequence of road segments towards the final destination point. In the higher-level route planning, a high level of abstraction is employed such that lower-level variables in regard to motion planning (regulatory rules of the road, presence of static and dynamic obstacles, vulnerable road users, more detailed and realistic road structure) are abstracted away from the problem instance, as also explained in [159]. Herewith, only higher-level metrics are considered, including the distance of the roads, current and projected traffic congestion on the roads, speed limits on the roads, direction of the traffic flow on each road, and estimated network quality index associated to each road. As another research dimension, the lower level and shorter-term planning problems of this hierarchy can be concerned by means of including lower-level variables, objectives, and constraints into the problem model. There could be two main approaches towards handling the overarching motion planning problem of autonomous vehicles [159]. On one hand, if the motion planning problem is considered as one optimization problem and solved at once, the most optimum solution can be reached in a computationally expensive way. As another option, if the motion planning problem is decomposed into sub-problems with relevant objectives and constraints associated to each layer, a sub-optimal solution can be reached in near real-time. Thus, studying this threshold might be a research topic in this context. Additionally, building a more comprehensive Local Dynamic Map (LDM) structure encompassing the information of various static and dynamic traffic participants on the urban road environment could be another relevant research problem. In this respect, the predictive, collision-free path planning and control of the autonomous vehicles at the lower layer by considering the forecasted trajectories of other traffic agents (vehicles, cyclists, pedestrians, etc.) could be a valuable research dimension.

Bibliography

[1] '3gpp ts 22.186 - enhancement of 3gpp support for v2x scenarios', 3rd Generation Partnership Project (3GPP), Tech. Rep., 2018.

[2] F. Sivrikaya, M. A. Khan, C. Bila, and S. Albayrak, 'Reciprocal impact of autonomous vehicles and network resource management', in *2017 IEEE Vehicular Networking Conference (VNC)*, 2017, pp. 231–234.

[3] *Etsi*, `https://www.etsi.org/`, Nov. 2020.

[4] *Cen*, `https://www.cen.eu/Pages/default.aspx`, Nov. 2020.

[5] *Ietf*, `https://www.ietf.org/`, Nov. 2020.

[6] *Iso*, `https://www.iso.org/home.html`, Nov. 2020.

[7] *Car 2 car communication consortium*, `https://www.car-2-car.org/index.php?id=1`, Nov. 2020.

[8] T. Kosch, I. Kulp, M. Bechler, M. Strassberger, B. Weyl, and R. Lasowski, 'Communication architecture for cooperative systems in europe', *IEEE Communications Magazine*, vol. 47, no. 5, pp. 116–125, 2009.

[9] A. Festag, 'Cooperative intelligent transport systems standards in europe', *IEEE Communications Magazine*, vol. 52, no. 12, pp. 166–172, 2014.

[10] *Safespot*, `http://www.safespot-eu.org/`, Nov. 2020.

[11] *Cvis*, `http://www.cvisproject.org/`, Nov. 2020.

[12] *Drive c2x*, `https://www.fokus.fraunhofer.de/`, Nov. 2020.

[13] 'Ieee guide for wireless access in vehicular environments (wave) architecture', IEEE Standards Association, Tech. Rep., 2014.

[14] 'Intelligent transport systems (its), communications architecture', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2010.

[15] 'Iso 21217:2014 intelligent transport systems - communications access for land mobiles (calm) - architecture', International Organization for Standardization (ISO), Tech. Rep., 2014.

[16] 'Intelligent transport systems (its); radiocommunications equipment operating in the 5 855 mhz to 5 925 mhz frequency band; harmonized standard covering the essential requirements of article 3.2 of directive 2014/53/eu', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2017.

[17] M. Karoui, A. Freitas, and G. Chalhoub, 'Performance comparison between lte-v2x and its-g5 under realistic urban scenarios', May 2020. DOI: `10.1109/VTC2020-Spring48590.2020.9129423`.

[18] 'Intelligent transport systems (its); decentralized congestion control mechanisms for intelligent transport systems operating in the 5 ghz range; access layer part', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2018.

[19] N. Lyamin, A. Vinel, D. Smely, and B. Bellalta, 'Etsi dcc: Decentralized congestion control in c-its', *IEEE Communications Magazine*, vol. 56, no. 12, pp. 112–118, 2018.

[20] 'Ieee standard for information technology - telecommunications and information exchange between systems local and metropolitan area networks - specific requirements - part 11 wireless lan medium access control (mac) and physical layer (phy) specifications', IEEE, Tech. Rep., 2016.

[21] E. G. Strom, 'On medium access and physical layer standards for cooperative intelligent transport systems in europe', *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1183–1188, 2011.

[22] 'Intelligent transport systems (its); vehicular communications; geonetworking; part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; sub-part 1: Media-independent functionality', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2020.

[23] 'Intelligent transport systems (its); vehicular communications; geographical area definition', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2011.

[24] S. Kuhlmorgen, I. Llatser, A. Festag, and G. Fettweis, 'Performance evaluation of etsi geonetworking for vehicular ad hoc networks', in *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, 2015, pp. 1–6.

[25] 'Intelligent transport systems (its); vehicular communications; geonetworking; part 5: Transport protocols; sub-part 1: Basic transport protocol', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2019.

[26] 'Intelligent transport systems (its); vehicular communications; geonetworking; part 6: Internet integration; sub-part 1: Transmission of ipv6 packets over geonetworking protocols', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2011.

[27] 'Intelligent transport systems (its); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2011.

[28] 'Intelligent transport systems (its); vehicular communications; basic set of applications; part 3: Specifications of decentralized environmental notification basic service', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2010.

[29] K. Sjoberg, P. Andres, T. Buburuzan, and A. Brakemeier, 'Cooperative intelligent transport systems in europe: Current deployment status and outlook', *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 89–97, 2017.

[30] 'Intelligent transport systems (its); vehicular communications; basic set of applications; part 3: Specifications of decentralized environmental notification basic service', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2014.

[31] 'Intelligent transport systems (its); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2014.

[32] 'Intelligent transport systems (its); users and applications requirements; part 2: Applications and facilities layer common data dictionary', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2018.

[33] 'Intelligent transport systems (its); vehicular communications; basic set of applications; definitions', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2009.

[34] *3gpp - about 3gpp home*, `https://www.3gpp.org/about-3gpp/about-3gpp`, Nov. 2020.

[35] *5gaa*, `https://5gaa.org/`, Nov. 2020.

[36] '3gpp ts 22.185 service requirements for v2x services; release 15', 3rd Generation Partnership Project (3GPP), Tech. Rep., 2018.

[37] X. Wang, S. Mao, and M. Gong, 'An overview of 3gpp cellular vehicle-to-everything standards', *GetMobile: Mobile Computing and Communications*, vol. 21, pp. 19–25, Nov. 2017. DOI: `10.1145/3161587.3161593`.

[38] S. Husain, A. Kunz, A. Prasad, E. Pateromichelakis, K. Samdanis, and J. Song, 'The road to 5g v2x: Ultra-high reliable communications', in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2018, pp. 1–6.

[39] R. Molina-Masegosa and J. Gozalvez, 'Lte-v for sidelink 5g v2x vehicular communications: A new 5g technology for short-range vehicle-to-everything communications', *IEEE Vehicular Technology Magazine*, vol. 12, no. 4, pp. 30–39, 2017.

[40] G. Naik, B. Choudhury, and J. Park, 'Ieee 802.11bd 5g nr v2x: Evolution of radio access technologies for v2x communications', *IEEE Access*, vol. 7, pp. 70 169–70 184, 2019.

[41] '3gpp tr 22.886 - study on enhancement of 3gpp support for 5g v2x services (release 15)', 3rd Generation Partnership Project (3GPP), Tech. Rep., 2017.

[42] V. Vukadinovic, K. Bakowski, P. Marsch, I. Garcia, H. Xu, M. Sybis, P. Sroka, K. Wesolowski, DavidLister, and I. Thibault, '3gpp c-v2x and ieee 802.11p for vehicle-to-vehicle communications in highway platooning scenarios', *Ad Hoc Networks*, vol. 74, Mar. 2018. DOI: `10.1016/j.adhoc.2018.03.004`.

[43] B. Netten, L. Kester, H. Wedemeijer, I.Passchier, and B. Driessen, 'Dynamap: A dynamic map for road side its stations', *Proceedings of the 20th ITS World Congress*, pp. 102–112, 2013.

[44] 'Intelligent transport systems (its); vehicular communications; basic set of applications; local dynamic map (ldm)', European Telecommunications Standards Institute (ETSI), Tech. Rep., 2014.

[45] 'Intelligent transport systems; extension of map database specifications for local dynamic map for applications of cooperative its', ISO/TS 17931, Tech. Rep., 2013.

[46] 'Intelligent transport systems; cooperative systems; definition of a global concept for local dynamic maps', ISO/TS 18750, Tech. Rep., 2015.

[47] N. Varga, L. Bokor, and H. Fischer, 'Ldm-based dynamic network discovery and selection for ipv6 mobility management optimization in c-its environments', in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, 2015, pp. 483–490.

[48] 'Safespot core architecture, ldm api and usage reference', 2010.

[49] H. Shimada, A. Yamaguchi, H. Takada, and K. Sato, 'Implementation and evaluation of local dynamic map in safety driving systems', *Journal of Transportation Technologies*, pp. 102–112, Mar. 2015. DOI: 10.4236/jtts.2015.52010.

[50] M. Maiouak and T. Taleb, 'Dynamic maps for automated driving and uav geofencing', *IEEE Wireless Communications*, vol. 26, no. 4, pp. 54–59, Aug. 2019. DOI: 10.1109/MWC.2019.1800544.

[51] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, 'A survey of motion planning and control techniques for self-driving urban vehicles', *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016, ISSN: 2379-8858. DOI: 10.1109/TIV.2016.2578706.

[52] J. Keiser, N. Masuch, M. L¸tzenberger, D. Grunewald, M. Kern, F. Trollmann, E. Acar, ≪ A. Salma, X. Dang, C. Kuster, and S. Albayrak, 'Ima - an adaptable and dynamic service platform for intermodal mobility assistance', in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2014, pp. 1521–1528. DOI: 10.1109/ITSC.2014.6957649.

[53] W. Goodall, T. D. Fishman, J. Bornstein, B. Bonthron, and T. Daberko, 'The rise of mobility as a service - reshaping how urbanities get around', Tech. Rep., Mar. 2018.

[54] *World's first driverless taxi system comes to singapore*, https://www.edb.gov.sg/en/news-and-events/insights/innovation/world-s-first-driverless-taxi-system-comes-to-singapore.html, Nov. 2020.

[55] 'The future of mobility is at our doorstep', McKinsey, Tech. Rep., 2020.

[56] J. Alonso-Mora, A. Wallar, and D. Rus, 'Predictive routing for autonomous mobility-on-demand systems with ride-sharing', in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 3583–3590. DOI: 10.1109/IROS.2017.8206203.

[57] Marczuk, Katarzyna A., Soh, Harold S.H., Azevedo, Carlos M.L., Lee, Der-Horng, and Frazzoli, Emilio, 'Simulation framework for rebalancing of autonomous mobility on demand systems', *MATEC Web Conf.*, vol. 81, p. 01 005, 2016. DOI: 10.1051/matecconf/20168101005. [Online]. Available: https://doi.org/10.1051/matecconf/20168101005.

[58] J. Pan, M. A. Khan, I. S. Popa, K. Zeitouni, and C. Borcea, 'Proactive vehicle re-routing strategies for congestion avoidance', in *2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, 2012, pp. 265–272.

[59] *Simulation of urban mobility*, https://www.eclipse.org/sumo/, Mar. 2020.

[60] A. Wegener, M. Piorkowski, M. Raya, H. Hellbruck, S. Fischer, and J. Hubaux, 'Traci: An interface for coupling road traffic and network simulators', in *ResearchGate*, Apr. 2008. DOI: 10.1145/1400713.1400740.

[61] N. S. Nafi, R. H. Khan, J. Y. Khan, and M. Gregory, 'A predictive road traffic management system based on vehicular ad-hoc network', in *2014 Australasian Telecommunication Networks and Applications Conference (ATNAC)*, Nov. 2014, pp. 135–140. DOI: 10.1109/ATNAC.2014.7020887.

[62] J. Wang and H. Niu, 'A distributed dynamic route guidance approach based on short-term forecasts in cooperative infrastructure-vehicle systems', *Transportation Research Part D Transport and Environment*, vol. 66, pp. 23–34, Jan. 2019.

[63] 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)*, Jun. 2016, pp. 726–731. DOI: 10.1109/ISCC.2016.7543822.

[64] J. Jeong, H. Jeong, E. Lee, T. Oh, and D. H. C. Du, 'Saint: Self-adaptive interactive navigation tool for cloud-based vehicular traffic optimization', *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 4053–4067, Jun. 2016, ISSN: 0018-9545. DOI: 10.1109/TVT.2015.2476958.

[65] S. Kraus, R. Parshani, and Y. Shavitt, 'A study on gossiping in transportation networks', *Vehicular Technology, IEEE Transactions on*, vol. 57, pp. 2602–2607, Aug. 2008. DOI: 10.1109/TVT.2007.912339.

[66] J. W. Wedel, B. Sch¸nemann, and I. Radusch, 'V2x-based traffic congestion recognition and avoidance', in *2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, Dec. 2009, pp. 637–641. DOI: 10.1109/I-SPAN.2009.71.

[67] J. W. Ding, C. F. Wang, F. H. Meng, and T. Y. Wu, 'Real-time vehicle route guidance using vehicle-to-vehicle communication', *IET Communications*, vol. 4, no. 7, pp. 870–883, Apr. 2010, ISSN: 1751-8628. DOI: 10.1049/iet-com.2009.0163.

[68] R. Claes, T. Holvoet, and D. Weyns, 'A decentralized approach for anticipatory vehicle routing using delegate multiagent systems', *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 364–373, Jun. 2011, ISSN: 1524-9050. DOI: 10.1109/TITS.2011.2105867.

[69] J. Lin, W. Yu, X. Yang, Q. Yang, X. Fu, and W. Zhao, 'A real-time en-route route guidance decision scheme for transportation-based cyberphysical systems', *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2551–2566, Mar. 2017, ISSN: 0018-9545. DOI: 10.1109/TVT.2016.2572123.

[70] I. Leontiadis, G. Marfia, D. Mack, G. Pau, C. Mascolo, and M. Gerla, 'On the effectiveness of an opportunistic traffic management system for vehicular networks', *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1537–1548, Dec. 2011, ISSN: 1524-9050. DOI: 10.1109/TITS.2011.2161469.

[71] 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*, vol. 64, no. 5, pp. 1664–1678, May 2015, ISSN: 0018-9545. DOI: 10.1109/TVT.2014.2335201.

[72] C. Brennand, A. Souza, G. Maia, A. Boukerche, H. Ramos, A. Loureiro, and L. Villas, 'An intelligent transportation system for detection and control of congested roads in urban centers', Jun. 2015. DOI: 10.1109/ISCC.2015.7405590.

[73] 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 *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, 2015, pp. 497–503.

[74] J. Pan, I. S. Popa, and C. Borcea, 'Divert: A distributed vehicular traffic rerouting system for congestion avoidance', *IEEE Transactions on Mobile Computing*, vol. 16, no. 1, pp. 58–72, 2017.

[75] A. Souza, R. Yokoyama, A. Boukerche, G. Maia, E. Cerqueira, A. Loureiro, and L. Villas, 'Icarus: Improvement of traffic condition through an alerting and rerouting system', *Computer Networks*, vol. 110, Sep. 2016. DOI: 10.1016/j.comnet.2016.09.011.

[76] W. Zhang, N. Aung, S. Dhelim, and Y. Ai, 'Diftos: A distributed infrastructure-free traffic optimization system based on vehicular ad hoc networks for urban environments', *Sensors*, vol. 18, Aug. 2018. DOI: 10.3390/s18082567.

[77] M. Ahmed, S. Iqbal, K. Awan, K. Sattar, Z. A. Khan, and H. Sherazi, 'A congestion aware route suggestion protocol for traffic management in internet of vehicles', *Arabian Journal for Science and Engineering*, Aug. 2019. DOI: 10.1007/s13369-019-04099-9.

[78] R. Doolan and G. Muntean, 'Ecotrec a novel vanet-based approach to reducing vehicle emissions', *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 3, pp. 608–620, 2017.

[79] 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)*, 2018, pp. 1–7.

[80] 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*, pp. 1–17, 2019.

[81] A. Mabrouk, A. Kobbane, M. EL Koutbi, and E. Sabir, 'Achieving always best connected service for a mobile user in vanet', in *2014 5th Workshop on Codes, Cryptography and Communication Systems (WCCCS)*, 2014, pp. 64–66.

[82] T. Hultman, A. Boudjadar, and M. Asplund, 'Connectivity-optimal shortest paths using crowdsourced data', in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 2016, pp. 1–6.

[83] *Openstreetmap*, `https://www.openstreetmap.de/`, Mar. 2021.

[84] *Geofabrik*, `http://www.geofabrik.de/`, Mar. 2021.

[85] *Osmconvert*, `https://wiki.openstreetmap.org/wiki/Osmconvert`, Mar. 2021.

[86] *Omnet++ simulation manual*, `https://doc.omnetpp.org/omnetpp/manual/`, Mar. 2021.

[87] *Pymoo - multi-objective optimization in python*, `https://pymoo.org/index.html`, Mar. 2021.

[88] T. Afrin and N. Yodo, 'A survey of road traffic congestion measures towards a sustainable and resilient transportation system', *Sustainability*, vol. 12, p. 4660, Jun. 2020. DOI: `10.3390/su12114660`.

[89] 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)*, 2018, pp. 2905–2910.

[90] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. Suganthan, and Q. Zhang, 'Multiobjective evolutionary algorithms: A survey of the state of the art', *Swarm and Evolutionary Computation*, vol. 1, pp. 32–49, Mar. 2011. DOI: `10.1016/j.swevo.2011.03.001`.

[91] G. Chiandussi, M. Codegone, S. Ferrero, and F. Varesio, 'Comparison of multiobjective optimization methodologies for engineering applications', *Computers and Mathematics with Applications*, vol. 63, pp. 912–942, Mar. 2012. DOI: `10.1016/j.camwa.2011.11.057`.

[92] C. Von Lücken, B. Baran, and C. Brizuela, 'A survey on multi-objective evolutionary algorithms for many-objective problems', *Computational Optimization and Applications*, pp. 1–50, Jul. 2014. DOI: 10.1007/s10589-014-9644-1.

[93] B. Qu, Y. Zhu, Y. Jiao, M. Wu, P. Suganthan, and J. Liang, 'A survey on multi-objective evolutionary algorithms for the solution of the environmental/economic dispatch problems', *Swarm and Evolutionary Computation*, Jun. 2017. DOI: 10.1016/j.swevo.2017.06.002.

[94] J. Zhang and L. Xing, 'A survey of multiobjective evolutionary algorithms', in *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, vol. 1, Jul. 2017, pp. 93–100. DOI: 10.1109/CSE-EUC.2017.27.

[95] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, 'A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms', *Soft Computing*, Dec. 2017. DOI: 10.1007/s00500-017-2965-0.

[96] W. Stadtler, 'A survey of multicriteria optimization or the vector maximum problem, part i: 1776 to 1960', vol. 29, 1979, pp. 1–52. DOI: 10.1007/BF00932634.

[97] E. Zitzler, M. Laumanns, and S. Bleuler, 'A tutorial on evolutionary multiobjective optimization', vol. 535, Jan. 2003. DOI: 10.1007/978-3-642-17144-4_1.

[98] T. Back, U. Hammel, and H. .-.-. Schwefel, 'Evolutionary computation: Comments on the history and current state', *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 3–17, 1997. DOI: 10.1109/4235.585888.

[99] P. A. Vikhar, 'Evolutionary algorithms: A critical review and its future prospects', in *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, 2016, pp. 261–265. DOI: 10.1109/ICGTSPICC.2016.7955308.

[100] A. G. Najera, 'Multi-objective evolutionary algorithms for vehicle routing problems', PhD thesis, The University of Birmingham, 2010.

[101] H. Kusetogullari, 'Network routing optimisation and effective multimedia transmission to enhance qos in communication networks', PhD thesis, The University of Warwick, 2012.

[102] *Genetic algorithm*, https://en.wikipedia.org/wiki/Genetic_algorithm, Mar. 2021.

[103] D. Beasley, D. R. Bull, and R. R. Martin, 'An overview of genetic algorithms part 1 fundamentals', *University Computing*, 1993.

[104] *Genetic algorithm options*, https://www.mathworks.com/help/gads/genetic-algorithm-options.html, Mar. 2021.

[105] M. Mitchell, 'Genetic algorithms an overview', *An Introduction to Genetic Algorithms, MIT Press*, vol. 1, Sep. 1995.

[106] Chang Wook Ahn and R. S. Ramakrishna, 'A genetic algorithm for shortest path routing problem and the sizing of populations', *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 566–579, 2002. DOI: 10.1109/TEVC.2002.804323.

[107] *Genetic algorithm options*, https://www.mathworks.com/help/gads/genetic-algorithm-options.html, Mar. 2021.

[108] O. Abdoun and J. Abouchabaka, 'A comparative study of adaptive crossover operators for genetic algorithms to resolve the traveling salesman problem', *CoRR*, vol. abs/1203.3097, 2012. arXiv: 1203.3097. [Online]. Available: http://arxiv.org/abs/1203.3097.

[109] *Genetic algorithms - selection*, https://medium.com/datadriveninvestor/genetic-algorithms-selection-5634cfc45d78, Mar. 2021.

[110] K. De Jong and W. Spears, 'A formal analysis of the role of multipoint crossover in genetic algorithms', *Annals of Mathematics and Artificial Intelligence - AMAI*, vol. 5, pp. 1–26, Mar. 1992. DOI: 10.1007/BF01530777.

[111] P. Kora and P. Yadlapalli, 'Crossover operators in genetic algorithms: A review', *International Journal of Computer Applications*, vol. 162, pp. 34–36, Mar. 2017. DOI: 10.5120/ijca2017913370.

[112] A. Chipperfield, P. Fleming, H. Pohlheim, and C. Fonseca, 'Genetic algorithm toolbox - for use with matlab', Tech. Rep., Mar. 2021.

[113] J. Mccall, 'Genetic algorithms for modelling and optimisation', *Journal of Computational and Applied Mathematics*, vol. 184, pp. 205–222, Dec. 2005. DOI: 10.1016/j.cam.2004.07.034.

[114] A. Otman and J. Abouchabaka, 'A comparative study of adaptive crossover operators for genetic algorithms to resolve the traveling salesman problem', vol. 31, Mar. 2012.

[115] M. Gen, Runwei Cheng, and Dingwei Wang, 'Genetic algorithms for solving shortest path problems', in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*, 1997, pp. 401–406. DOI: 10.1109/ICEC.1997.592343.

[116] H. Kusetogullari, M. Leeson, B. Kole, and E. Hines, 'Meta-heuristic algorithms for optimized network flow wavelet-based image coding', *Applied Soft Computing*, vol. 14, pp. 536–553, Jan. 2014. DOI: 10.1016/j.asoc.2013.09.001.

[117] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, 2001, ISBN: 978-0-471-87339-6.

[118] M. A. Nayeem, M. M. Islam, and X. Yao, 'Solving transit network design problem using many-objective evolutionary approach', *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3952–3963, 2019. DOI: 10.1109/TITS.2018.2883511.

[119] J. Schaffer, 'Multiple objective optimization with vector evaluated genetic algorithms.', Jan. 1985, pp. 93–100.

[120] C. Coello, G. B. Lamont, and D. A. van Veldhuizen, *Evoluionary Algorithms for Solving Multi-Objective Problems*. Springer Science and Business Media, 2007, ISBN: 978-0-387-33254-3.

[121] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, 'A fast and elitist multiobjective genetic algorithm: Nsga-ii', *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr. 2002, ISSN: 1941-0026. DOI: 10.1109/4235.996017.

[122] N. Srinivas and K. Deb, 'Multiobjective function optimization using nondominated sorting genetic algorithms', vol. 2, Jun. 2000.

[123] *Nsga-ii optimization: Understand fast how it works [complete explanation]*, https://www.youtube.com/watch?v=SL-u_7hIqjA&t=675s, Mar. 2021.

[124] F. J. Martinez, C. K. Toh, J. C. Cano, C. T. Calafate, and P. Manzoni, 'A survey and comparative study of simulators for vehicular ad hoc networks (vanets)', in *WIRELESS COMMUNICATIONS AND MOBILE COMPUTING*, 2011. DOI: 10.1002/wcm.859.

[125] Y. Hou, Y. Zhao, A. Wagh, L. Zhang, C. Qiao, K. F. Hulme, C. Wu, A. W. Sadek, and X. Liu, 'Simulation-based testing and evaluation tools for transportation cyber?physical systems', *IEEE Transactions on Vehicular Technology*, vol. 65, no. 3, pp. 1098–1108, Mar. 2016, ISSN: 0018-9545. DOI: 10.1109/TVT.2015.2407614.

[126] C. Sommer, J. H?rri, F. Hrizi, B. Sch¸nemann, and F. Dressler, 'Simulation tools and techniques for vehicular communications and applications', *Vehicular ad hoc Networks*, pp. 365–392, 2015.

[127] M. A. Hoque, X. Hong, and S. Ahmed, 'Parallel closed-loop connected vehicle simulator for large-scale transportation network management: Challenges, issues, and solution approaches', *IEEE Intelligent Transportation Systems Magazine*, pp. 1–1, 2018, ISSN: 1939-1390. DOI: 10.1109/MITS.2018.2879163.

[128] J. Yin, T. ElBatt, G. Y. B. Ryu, S. Habermas, H. Krishnan, and T. Talty, 'Performance evaluation of safety applications over dsrc vehicular ad hoc networks', in *VANET*, Jan. 2004.

[129] F. K. Karnadi, Z. H. Mo, and K. Lan, 'Rapid generation of realistic mobility models for vanet', in *2007 IEEE Wireless Communications and Networking Conference*, Mar. 2007, pp. 2506–2511. DOI: 10.1109/WCNC.2007.467.

[130] J. Harri, F. Filali, C. Bonnet, and M. Fiore, 'Vanetmobisim: Generating realistic mobility patterns for vanets', in *VANET'06*, Sep. 2006.

[131] A. Hassan and T. Larsson, 'On the requirements on models and simulator design for integrated vanet simulation', in *8th International Workshop on Intelligent Transportation (WIT 2011)*, Mar. 2011, pp. 191–196.

[132] D. R. Choffnes and F. E. Bustamante, 'An integrated mobility and traffic model for vehicular wireless networks', in *VANET'05*, Sep. 2005.

[133] R. Mangharam, D. S. Weller, D. D. Stancil, R. Rajkumar, and J. S. Parikh, 'Groovesim: A topography-accurate simulator for geographic routing in vehicular networks', in *VANET'05*, Sep. 2005.

[134] K. Ibrahim and M. C. Weigle, 'Ash: Application-aware swans with highway mobility', in *IEEE INFOCOM Workshops 2008*, Apr. 2008, pp. 1–6. DOI: 10.1109/INFOCOM.2008.4544652.

[135] R. Mangharam, D. Weller, R. Rajkumar, P. Mudalige, and F. Bai, 'Groovenet: A hybrid simulator for vehicle-to-vehicle networks', in *2006 3rd Annual International Conference on Mobile and Ubiquitous Systems - Workshops*, Jul. 2006, pp. 1–8. DOI: 10.1109/MOBIQW.2006.361773.

[136] M. Piurkowski, M. Raya, A. L. Lugo, P. Papadimitratos, M. Grossglauser, and J. Hubaux, 'Trans realistic joint traffic and network simulator for vanets', in *ACM SIGMOBILE Mobile Computing and Communications Review)*, vol. 12, Jan. 2008, pp. 31–33.

[137] C. Sommer, R. German, and F. Dressler, 'Bidirectionally coupled network and road traffic simulation for improved ivc analysis', *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 3–15, Jan. 2011, ISSN: 1536-1233. DOI: 10.1109/TMC.2010.133.

[138] U. Dwivedi and A. R. Upadhyay, 'Vehicle to vehicle communication in vehicular network simulation environment: Analysis and future perspectives', in *Journal of Network Communications and Emerging Technologies (JNCET))*, vol. 8, Feb. 2018.

[139] S. Rakkesh, A. Weerasinghe, and R. Ranasinghe, 'Equipoising multi-modal traffic environments using vehicular ad-hoc networks', in *International Journal on Advances in ICT for Emerging Regions)*, Dec. 2017.

[140] J. Sun, Y. Yang, and K. Li, 'Integrated coupling of road traffic and network simulationfor realistic emulation ofconnected vehicle applications', in *Simulation: Transactions of the Society for Modeling and Simulation International*, 2016. DOI: 10.1177/0037549716634189.

[141] A. Choudhury, T. Maszczyk, C. B. Math, H. Li, and J. Dauwels, 'An integrated simulation environment for testing v2x protocols and applications', in *The International Conference on Computational Science*, vol. 80, 2016, pp. 2042–2052. DOI: 10.1016/j.procs.2016.05.524.

[142] *Itetris*, http://www.ict-itetris.eu/simulator/, Accessed: 2019-12-02.

[143] V. Kumar, L. Lin, D. Krajzewicz, F. Hrizi, O. Martinez, J. Gozalvez, and R. Bauza, 'Itetris: Adaptation of its technologies for large scale integrated simulation', in *2010 IEEE 71st Vehicular Technology Conference*, May 2010, pp. 1–5. DOI: 10.1109/VETECS.2010.5494182.

[144] *Vsimrti - smart mobility solution*, https://www.dcaiti.tu-berlin.de/research/simulation/, Accessed: 2019-31-01.

[145] B. Schunemann, 'V2x simulation runtime infrastructure vsimrti: An assessment tool to design smart traffic management systems', in *Elseiver, Computer Networks)*, May 2011. DOI: 10.1016/j.comnet.2011.05.005.

[146] 'Ieee standard for modeling and simulation (m and s) high level architecture (hla) federate interface specification', in *IEEE Computer Society*, Aug. 2010.

[147] *Veins - vehicles in network simulation*, https://veins.car2x.org/, Dec. 2020.

[148] C. Sommer, R. German, and F. Dressler, 'Bidirectionally coupled network and road traffic simulation for improved ivc analysis', *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 3–15, 2011.

[149] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. Haneveld, T. Parker, O. Visser, H. Lichte, and S. Valentin, 'Simulating wireless and mobile networks in omnet++ the mixim vision.', Jan. 2008, p. 71. DOI: 10.1145/1416222.1416302.

[150] *Tracl - traffic control interface*, https://sumo.dlr.de/docs/TraCI.html, Dec. 2020.

[151] D. Eckhoff, C. Sommer, and F. Dressler, 'On the necessity of accurate ieee 802.11p models for ivc protocol simulation', in *2012 IEEE 75th Vehicular Technology Conference (VTC Spring)*, 2012, pp. 1–5.

[152] P. Fuxjaeger, A. Costantini, D. Valerio, P. Castiglione, T. Zemen, and F. Ricciato, 'Ieee 802.11p transmission using gnuradio', Mar. 2010.

[153] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, 'Simulation of urban mobility - an overview', in *SIMUL 2011 : The Third International Conference on Advances in System Simulation*, 2011.

[154] A. Varga, 'The omnet++ discrete event simulation system', *Proc. ESM'2001*, vol. 9, Jan. 2001.

[155] *Xml-rpc server*, https://docs.python.org/3/library/xmlrpc.server.html, Dec. 2020.

[156] D. Kornek, M. Schack, E. Slottke, O. Klemp, I. Rolfes, and T. Kürner, 'Effects of antenna characteristics and placements on a vehicle-to-vehicle channel scenario', in *2010 IEEE International Conference on Communications Workshops*, 2010, pp. 1–5. DOI: 10.1109/ICCW.2010.5503935.

[157] *Dijkstra's shortest path with minimum edges*, https://www.geeksforgeeks.org/dijkstras-shortest-path-with-minimum-edges/, Dec. 2020.

[158] M. Cap and J. Alonso-Mora, 'Multi-objective analysis of ridesharing in automated mobility-on-demand', Jun. 2018. DOI: 10.15607/RSS.2018.XIV.039.

[159] *Motion planning for self-driving cars*, https://www.coursera.org/learn/motion-planning-self-driving-cars/home/welcome, Apr. 2021.