Multi-Agent Control of Urban Transportation Networks and of Hybrid Systems with Limited Information Sharing

Renshi Luo

Cover illustration: Inspired by the picture 'move002.jpg' on Mobile Robotics Laboratory Media Channel and designed by Yanchun Wei.

Multi-Agent Control of Urban Transportation Networks and of Hybrid Systems with Limited Information Sharing

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
woensdag 30 november 2016 om 10.00 uur

door

Renshi LUO,
Master of Science in Control Engineering, Beijing Jiaotong University,
geboren te Longzhou, Guangxi, China.

This dissertation has been approved by the
promotor: Prof. dr. ir. B. De Schutter
copromotor: Dr.ir A.J.J. van den Boom

Composition of the doctoral committee:

Rector Magnificus	chairman
Prof. dr. ir. B. De Schutter	Technische Universiteit Delft
Dr.ir. A.J.J. van den Boom	Technische Universiteit Delft

Independent members:

Prof. dr. ir. B. van Arem	Technische Universiteit Delft
Prof. dr. ir. E.C. van Berkum	Universiteit Twente
Prof. dr. ir. J. Hellendoorn	Technische Universiteit Delft
Prof. dr. P. Maestre	University of Seville, Spain

Other member:

Prof. dr. R. Bourdais	CentraleSupelec, France

Research described in this thesis was supported by the China Scholarship Council (CSC) under Grant 201207090001, Delft Center for Systems and Control, and by the Van Gogh project VGP.14/47.

TRAIL Thesis Series T2016/21, the Netherlands TRAIL Research School
P.O. Box 5017
2600 GA Delft, The Netherlands
T: +31 (0) 15 278 6046
T: +31 (0) 15 278 4333
E: info@rstrail.nl

Published and distributed by: Renshi Luo
E-mail: luobjtu@gmail.com

ISBN 978-90-5584-213-1

Keywords: multi-agent control, model predictive control, urban transportation networks, dynamic traffic routing, hybrid systems, limited information sharing

Printed in the Netherlands

Preface

By the time I started writing my own thesis, I have already collected quite a number of copies of Ph.D thesis of colleagues and friends. I would never forget the smiles of those doctors when they gave me the booklets describing their academic achievements. Now, after working intensively for four years, I manage to finish my Ph.D research and to collect contents for my own booklet. On this unforgettable journey, various people have offered me help in their unique ways. I own those kind people a lot of thanks.

First and foremost, I would like to thank my supervisors Prof.dr.ir. Bart De Schutter and Dr.ir. Ton van den Boom for their support and care. I would not have been able to write this thesis without their persistent guidance. I really appreciate that Bart always provides instant and accurate feedback on my work. Although he is continuously occupied by heavy work loads and live far away in Belgium, he is always available to offer help whenever I have doubts in my work and in my personal life. This sense of security provided by him comforted me greatly for all the times I felt overwhelmed by the demanding work and the distressing cultural difference. Besides, he never hold back his complements whenever I give a nice presentation, prepare a well-written report, show kindness to colleagues and visitors, etc. It is his appreciation that encourages me to learn, to grow, and to be a better myself. I am really grateful to Ton for his enthusiasm in my research and his kindness. He always come up with great ideas to push forward my research and is always willing to share many of his experiences in life. The support he gave me really helped me finish my Ph.D research. It has been a great pleasure for me to work with him.

I would also like to thank Prof. Romain Bourdais for providing me a rewarding opportunity, which many people as deserving as me did not get, to cooperate with him. His creative thinking, vast knowledge, and inspiring criticisms have contribute significantly to our joint work. Apart from helping me in work, I thank Romain for his kindness and his easily approachable character. He always came to me and greeted me with firm handshakes every day I worked in CentraleSupelec, Rennes. He accepted me as a close friend by constantly inviting me to lunches, dinners, drinks, and table football games. He even shared with me his passion at music and the songs he performed together with his wife and their lovely children. It has been an amazing experience to interact with him.

I thank my Ph.D committee members, Prof.dr.ir. Bart van Arem, Prof.dr.ir. Hans Hellendoorn, Prof.dr.ir. Eric van Berkum, Prof. Pepe Maestre, and Prof. Romain Bourdais, for their valuable time and constructing comments provided to help me improve this thesis.

I have a great time working at Delft Center for Systems and Control. I would like to thank Prof.dr.ir. Hans Hellendoorn, Kitty, Marieke, Heleen, Kiran, Ditske, Olaf, Saskia, and Esther for being helpful and friendly when I have questions regarding forms and financial matters. I thank Will for answering all my questions regarding use of DCSC Cluster. I thank Kim for being my only officemate and for all the sincere conversations we had during both good and

bad times. I thank Sachin, Edwin, Bart Karsbergen, Subramanya, Hans Verstraete, Laurens, Hildo, Elisabeth, Baptiste, and Paolo for inviting me to various social activities. I thank Yihui, Noortje, Mohammad, Yashar, and Patricio for offering me help in work. I thank Amir for supporting me when we visited CentraleSupelec together and for his generosity in paying for the beers. I thank Laura and her boyfriend for good memories of traveling with me in Japan. I thank Chengpu and his family, Anna, Anahita, Max, Marco, Farid, Yiming, Zhe, Yu, Yue, Jun, Le, Shuai liu, Jia, Anqi, Zhou, Shuai Yuan, Hai, Huijuan, Zhao and Shukai for their sincere friendship.

I enjoyed every moment I spent with the colleagues in CentraleSupelec, Rennes. I would like to thank Herve, Pierre, Jaehwa and Khang for their hospitality. I also would like to thank Claude and Manu, Yang, Alice, Lalie and Laurent for the happy ambiance during my stay in the Claude Family.

I have received abundant care and support from the loving Chinese Fellowship of Delft. I would like to thank uncle Didy and aunt Vera, Sukiato, T.K Choi and his family, Weiwei, Anyao and his family, Weishan, Liang and his family, Philip and Xinting, Fei, Yanchun, Yan and Tianmu, Ruoyang and Shasha, Ye, Yi, Zanni, Meixia, Yingzhu, Nelly, Wexin, Qianrong, Chang, Haiyan, Ou, Jingwen, Weiyuan, Yonghui, Siqi and Michelle, Anthony, Zihao, Xuanyu and Xuan, Anabin, Yiyun, Si, Sujuan, Xiaohui, Lianchao, Li, Yining, Dongdong and Muyan for their sincere love. My special gratitude goes to my girlfriend Lulu for her continuous support and encouragement.

Most importantly, I want to thank my beloved parents and my brother for their unconditional love and support. Although I left them at a very young age and went further and further, their constant love and care keeping reminding me that they are always with me. I am really grateful that I was born into such a loving family.

Finally, I would like to apologize to all the persons who contributed in one way or another to my work but are not mentioned here, and to thank them all together.

Renshi Luo

Delft, November 2016

Contents

Chapter 1

Introduction

In this chapter we first present the motivation for the research addressed in this thesis. After that, we indicate the scope of the research. Finally, we give a general overview of this thesis including the outline and the main contributions.

1.1 Motivation

As the population in urban areas is increasing, the demand for transportation services is also increasing rapidly. Since over 60% of the urban traffic is road traffic [110], the problems caused by the increasing demand of road traffic, such as large numbers of injuries and fatalities, frequent congestion, high levels of energy consumption and pollution, and high levels of noise, are getting more and more severe [39, 104, 118]. In order to satisfy the increasing demand for road transportation services and in order to mitigate the problems caused by increasing road traffic, effective and efficient traffic control strategies for urban transportation networks are urgently required. Urban traffic control consist of obtaining traffic information and applying control measures e.g., control of traffic lights and dynamic routing of vehicles. Conventionally, there is a traffic control center for each urban transportation network. However, an urban transportation network may consist of a large number of roads and intersections, which requires extensive communication efforts for transferring the states of the network to the traffic control center. Besides, the dynamics of traffic in the network are highly complicated, which requires extensive computation efforts for solving the resulting large-scale traffic control problem. For reasons of scalability and fast computation, a centralized control method will not be tractable for the control of large-scale urban transportation network of the future.

Except for control of large-scale transportation networks, control of multi-agent systems is also addressed in this thesis. Multi-agent systems, like transportation systems, manufacturing systems, power systems, financial systems, are composed of multiple subsystems with interactions [64]. Multi-agent systems research is facing a variety of challenges, of which a crucial one is to design mechanisms for coordinating agents that have limited information sharing with each other in order to protect confidential information of local subsystems while at the same time still aiming for global performance [43]. In order to achieve globally satisfactory performance, the agents need to assist each other to make better decisions about their actions. However, the cooperation among agents is made much more difficult when the individual agents have to regulate hybrid subsystems that contain both continuous components and discrete components. In fact, this will result

in having to solve mixed-integer programming problems in a distributed way, for which there has not yet been a successful algorithm.

1.2 Scope of the research

In this thesis we investigate and develop efficient solution methods for control of urban transportation networks and for control of multi-agent hybrid systems by employing state-of-the-art control methods and optimization techniques.

For control of transportation networks, dynamic traffic routing refers to the control process that influences the routes of traffic flows going through the network in response to the real-time traffic conditions of the network. In this thesis, we focus on solving the dynamic traffic routing problem for urban transportation networks by assuming all O-D (origin - destination) traffic demands to be given. We aim to develop efficient solution methods for the dynamic traffic routing problem so that the total travel cost including e.g. the total time spent and the total energy consumption is minimized.

For control of multi-agent hybrid systems, we focus on a class of hybrid systems that are governed by discrete inputs and that are subject to global hard constraints. In particular, each subsystem is characterized by a convex objective function and a strictly increasing constraint function with respect to the local control variable. Besides, each subsystem only shares limited information with the external environment. We aim to develop a multi-agent model predictive control method for such a class of hybrid systems based on a distributed optimization algorithm. We apply the developed multi-agent control method to the charging control of a fleet of electric vehicles.

Note that the control approaches that we develop in this thesis can similarly be extended to other applications, e.g., control of power distribution networks and energy management for smart buildings.

1.3 General overview of the thesis

1.3.1 Outline of the thesis

This thesis is divided into two parts. In the first part, we address the dynamic traffic routing problem for urban transportation networks. In the second part, we address multi-agent model predictive control of a class of hybrid systems with limited information sharing and subject to global hard constraints. More specifically, Figure 1 illustrates the organization and the relation between different parts and chapters of the thesis.

First, Chapter 2 gives an overview of the two main research topics i.e., control of urban transportation networks and multi-agent control of hybrid systems. More specifically, we present the surveys on the two main research topics, including the existing approaches, challenges, and future trends.

After that, in Chapter 3 and Chapter 4, we focus on multi-agent dynamic routing of cybercars in a cybernetic transportation network (i.e. a road network only open to cybercars). More specifically, Chapter 3 presents a discrete-time and a discrete-event model of the dynamics and the energy consumption of the whole fleet of cybercars in the network. Besides, the main features of the two modeling methods are discussed. For the sake of simplicity and fast computation, the discrete-time modeling is selected for controller

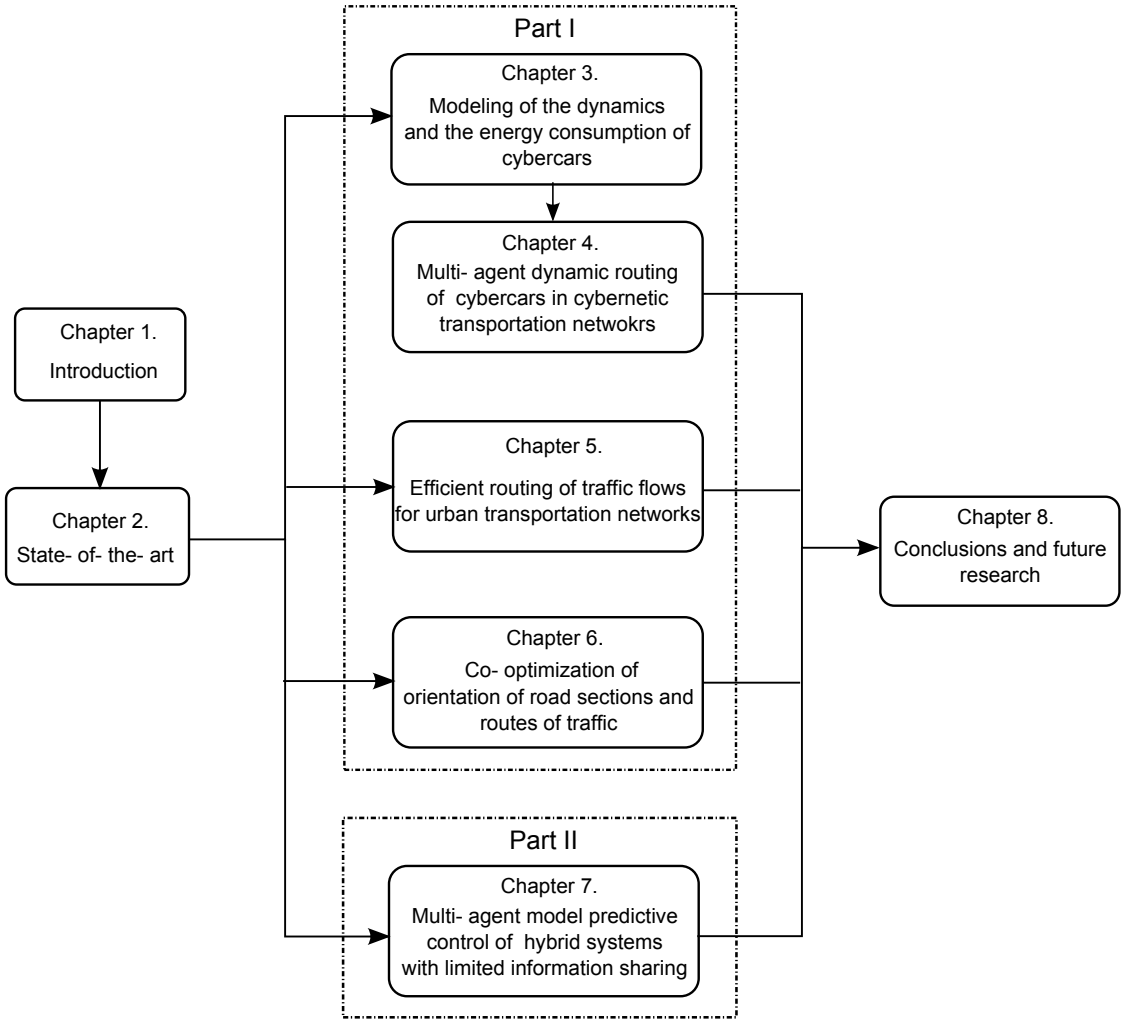

Figure 1.1: Structure of the thesis

design in the reminder of the thesis. In Chapter 4 we propose several multi-agent control methods including multi-agent model predictive control and parameterized control for dynamic routing of cybercars based on the discrete-time model of the dynamics and the energy consumption of cybercars.

Next, in Chapter 5, we address the dynamic traffic routing problem for urban transportation networks considering dynamics of traffic flows. We aim to minimize the total travel cost of all flows including the total time spent and the total energy consumption. In order to reach a well-balanced trade-off between the quality of solutions and the computation costs, we propose two novel multi-agent control approaches for dynamic traffic routing. First, the dynamic traffic routing problem is addressed based on network division and a hierarchical control method is proposed. After that, the problem is addressed based on merging nodes and links and a bi-level control method is proposed.

After that, Chapter 6 addresses the co-optimization problem of jointly determining the orientation of urban road sections and the routes of traffic flows. More specifically, we assume that the orientation of each road section in an urban transportation network can be changed in each control period and focus on the co-optimization problem that jointly determines the orientation of road sections in the network and the routes of traffic flows. By assuming circular orientation of traffic flow in each elementary cycle of roads in the

network and modeling the relation between the orientation of road sections and the circular orientation of traffic flows in the elementary cycles, the number of binary variables involved in the optimization problem is substantially reduced with respect to considering the orientation of each road section independently.

In Chapter 7, we propose a multi-agent model predictive control method for a class of hybrid systems governed by discrete inputs and subject to global hard constraints. The proposed multi-agent control method is based on a distributed resource allocation coordination algorithm and only requires limited information sharing from the local control problems of the subsystems. Thanks to primal decomposition of the global constraints, the distributed algorithm can always guarantee global feasibility of local control decisions, even when stopped prematurely. Besides, since the control variables are not continuous but discrete, a mechanism is developed to branch the overall solution space based on the outcome of the resource allocation coordination algorithm at each node of the search tree.

Finally, the thesis is concluded in Chapter 8, where the general summary of the thesis is presented. Besides, several possible directions for future work are recommended.

Note that most of the material included in this thesis has already been published or is under review. In particular, Chapter 3 is based on [76] and [81], Chapter 4 is based on [81] and [82], Chapters 5 and 6 are based on [80], and Chapter 7 is based on [77–79].

1.3.2 Main contributions

The main contributions of the thesis with respect to urban traffic routing and multi-agent control of hybrid systems are the following:

Urban traffic routing

- We develop two models for the dynamics and the energy consumption of cybercars in a road network that is only open to cybercars, and propose several tractable and scalable multi-agent control methods to solve the dynamic routing problem of cybercars.
- We propose two novel solution methods for dynamic traffic routing in urban transportation networks.
- We propose an efficient solution method for jointly optimizing the orientation of urban road sections and the routes of traffic flows.

Multi-agent control of hybrid systems

- We propose a novel multi-agent model predictive control for a class of hybrid systems subject to global constraints that requires only limited information sharing among local control agents and that guarantees the global feasibility of local control decisions.

Chapter 2

Traffic Management and Control of Hybrid Systems

In this chapter we present the background of the researches included in this thesis. The chapter is organized as follows. Firstly, in Section 2.1 we introduce the concepts of optimal control and model predictive control that are used both in traffic management and in control of hybrid systems. After that, the introduction to traffic management control and the introduction to control of hybrid systems are presented in Sections 2.2 and 2.3, respectively. Finally, we summarize the chapter in Section 2.4.

2.1 Optimal control and model predictive control

Optimal control and model predictive control are dynamic control methods that determine the control actions based on solving optimization problems. In this section we present the general description, including the concepts and the advantages and disadvantages, of the two control methods. Besides, we also present the algorithms that can be used to solve the resulting optimization problems in the two control methods.

2.1.1 Optimal control

Theoretical framework

Optimal control determines a sequence of admissible control actions for a given system over the entire control period by optimizing a performance function while satisfying the operational constraints of the system, see, e.g., [14, 67, 121]. However, in optimal control, the control inputs of the system are computed using only the measurements of the current state of the system and the model of the system. Hence, optimal control is an open-loop control approach that does not use any feedback. More specifically, a schematic diagram of optimal control is given in Figure 2.1. The essential elements involved in optimal control of a system are:

- a model of the system
- measurements of the initial state of the system
- an objective function

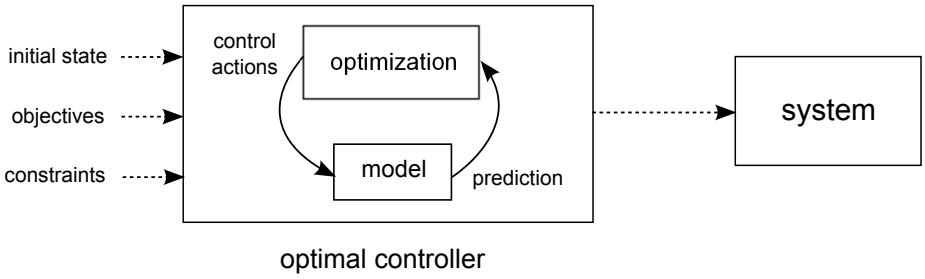

Figure 2.1: Optimal control

- operational constraints of the system

Generally, a standard discrete-time optimal control problem is formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & \sum_{k=0}^{k_{\text{end}}-1} J\Big(x(k),u(k),d(k)\Big) + P\Big(x(k_{\text{end}})\Big) && (2.1)\\ \text{s.t.} \quad & x(0) = x_0 \\ & x(k+1) = F_{\text{model}}\Big(x(k),u(k),d(k)\Big) \qquad \text{for } k = 0,1,...,k_{\text{end}}-1 \\ & G_{\text{constraint}}\Big(x(k),u(k),k\Big) \le 0 \qquad \text{for } k = 0,1,...,k_{\text{end}}-1 \end{aligned}$$

where $x(k) \in \mathbb{R}^n$ denotes the state of the system at time step k, $u(k) \in \mathbb{R}^m$ denotes the control input to the system at time step k, and $d(k) \in \mathbb{R}^h$ denotes the disturbances to the system at time step k. Besides, the functions $J(\cdot)$ and $P(\cdot)$ determine the cost of the system at each time step and the terminal penalty to the system respectively, and the functions $F_{\text{model}}(\cdot)$ and $G_{\text{constraint}}(\cdot)$ describe the dynamics of the system and the constraints imposed on the system respectively.

Therefore, for a given simulation period $[0, k_{\text{end}}T]$ with T denoting the sample time, the optimal control of a system consists in determining a sequence of control actions $\boldsymbol{u} = [u^{\text{T}}(0), u^{\text{T}}(1), ..., u^{\text{T}}(k_{\text{end}}-1)]^{\text{T}}$ by solving an optimization problem (2.1).

Advantages and disadvantages

The main advantage of optimal control is that it can be used to control multi-input multi-output dynamical systems and to handle the operational constraints of the systems explicitly. However, since optimal control is essentially an open-loop control method, the main drawback of optimal control is its performance loss or even incapability of satisfying constraints due to model mismatch errors and environmental disturbances.

Optimization algorithms

For some cases where the resulting optimization problems are convex [20] or if the solutions to the resulting optimization problems can be computed analytically [45, 101], the optimal control method is able to find the optimal sequence of control actions efficiently.

However, for many cases, the resulting optimization problem is nonlinear and nonsmooth, i.e. nonlinear programming problem, or even subject to constraint that the control inputs are restricted to integer values, i.e. mixed integer programming problem. Actually, those problems are nonconvex and are computationally hard to solve. Although

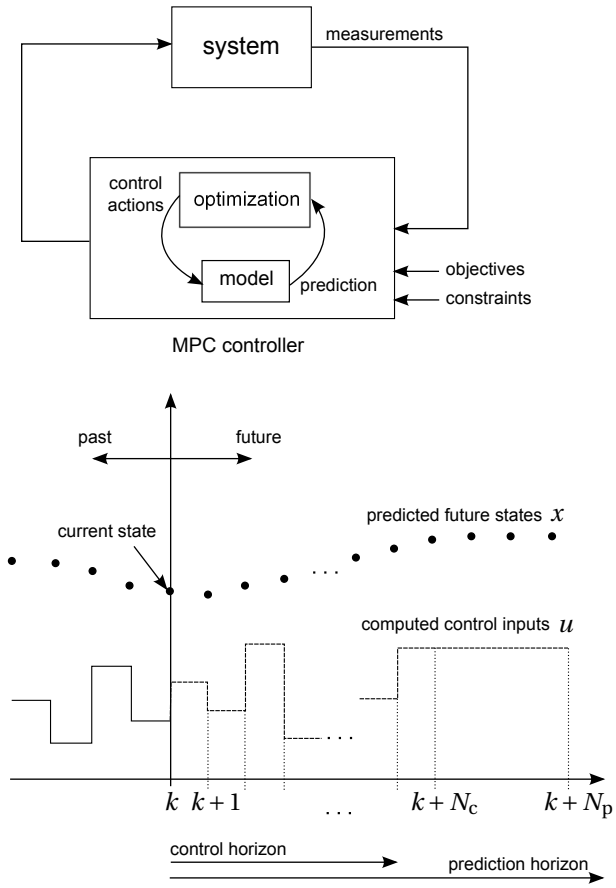

Figure 2.2: Model predictive control

there are several algorithms, e.g., multi-start sequential quadratic programming, multi-start simulated annealing, and multi-start pattern search [15], developed for solving nonlinear programming problem, and, e.g., genetic algorithm [100] and DIRECT [63], developed for solving mixed integer programming problem, these algorithms cannot guarantee that the global optimal solution is found.

2.1.2 Model predictive control

Theoretical framework

Model predictive control (MPC) has been widely recognized as a high-performance control approach for complex and constrained systems [22, 83]. In MPC, the control actions over a certain time span in the future are determined by solving a constrained optimization problem that includes the model of the system, the operational constraints, and the goal of control explicitly, in a receding horizon fashion.

Different from optimal control, MPC is a feedback control method that computes the control input to the system repeatedly by solving an on-line optimization problem at each control step. A schematic diagram of model predictive control is shown in Figure 2.2. More specifically, at each control step k, the MPC controller first makes a measurement of the current state $x(k)$ of the system. After that, the controller uses a prediction model of the system and on-line optimization to determine the optimal control actions over a given prediction period $\big[kT,(k+N_p)T\big)$, where N_p denotes the prediction horizon. One possible

way to reduce the computational complexity of the on-line optimization problem is to assume that the control inputs beyond the control horizon N_c become constant, i.e., $u(k+j) = u(k+j-1)$ for $j = N_c, ..., N_p - 1$. Therefore, the standard model prediction control problem is formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{u}} \ & \sum_{j=0}^{N_p-1} J\Big(x(k+j), u(k+j), d(k+j)\Big) + P\Big(x(k+N_p)\Big) && (2.2)\\ \text{s.t.} \ & x(k+j+1) = F_{\text{model}}\Big(x(k+j), u(k+j), d(k+j)\Big) && \text{for } j = 0, 1, ..., N_p - 1 \\ & G_{\text{constraint}}\Big(x(k+j), u(k+j), k+j\Big) \leq 0 && \text{for } j = 0, 1, ..., N_p - 1 \\ & u(k+j) = u(k+j-1) && \text{for } j = N_c, ..., N_p - 1 \end{aligned}$$

where the variables and the functions have already been defined in (2.1).

After the optimal control sequence $\boldsymbol{u}^* = \left[u^{*\mathrm{T}}(k), ..., u^{*\mathrm{T}}(k+N_c)\right]^{\mathrm{T}}$ are computed, only the first control action $u^*(k)$ is implemented on the real system, and subsequently the horizon is shifted. At the next control step, the new state of the system is measured, and a new optimization problem is solved using this new information. Recurrently, the receding horizon control procedure is repeated until the end of the overall control period.

Advantages and disadvantages

As optimal control, MPC can be used to control multi-input multi-output dynamical systems by taking into account the operational constraints of the system explicitly. Compared with optimal control, one advantage of MPC is the feedback mechanism which makes the system under the control of an MPC controller more robust to uncertainties and disturbances. Another advantage of MPC is that the on-line optimization problems are computationally less complex than the optimization problem involved in optimal control since N_c and N_p are smaller than k_{end}. However, this comes at the cost of loosing performance due to finite prediction and control horizons.

Optimization algorithms

The on-line optimization problems involved in MPC can be solved using the same optimization algorithms as for solving the optimization problems involved in optimal control.

Variants

So far we have presented the general concept of centralized MPC, where there is only one MPC controller controlling a whole system. However, when dealing with large-scale systems, centralized MPC would not be tractable due to the heavy communication burden in acquiring the state of the system and the high computational complexity in solving the resulting on-line optimization problems. Therefore, in order to deal with large-scale systems, several alternative MPC approaches have been developed, such as:

- Decentralized MPC
- Distributed MPC

- Hierarchical MPC
- Parameterized MPC

For the sake of simplicity, we only briefly describe the main idea of these alternative MPC approaches.

In decentralized MPC [65, 116], the whole system is divided into a group of subsystems that are independently controlled by local MPC controllers. More specifically, based on the local state and the prediction model of the corresponding subsystem, each local MPC controller determines the local control inputs to the subsystem by solving a local optimization problem. Compared with centralized MPC, the main advantage of MPC lies in the fact that the independent local optimization problems are much smaller and simpler, and hence are much easier to solve. However, this advantage of fast control typically comes at the cost of degraded overall performance.

Distributed MPC [23, 108] is an extension of decentralized MPC, where the local MPC controllers also exchange information regarding their future control actions while solving local optimization problems. More specifically, in distributed MPC, the local MPC controllers do not solve their local problems independently but cooperatively by exchanging information with each other to achieve global optimal performance. Compared with centralized and decentralized MPC, distributed MPC is typically able to achieve a well-balanced trade-off between performance and computation speed.

Hierarchical MPC [112, 113], which is also referred to as multi-level MPC, consists of multiple control levels. In a hierarchical MPC control set-up, the MPC controllers at higher levels have authority over the MPC controllers at lower levers, whereas the MPC controllers at the same level have equality authority relationships. More specifically, the MPC controllers at the high levels perform supervisory and strategic control of the system typically using a macroscopic model of the system, while the local MPC controllers at the lower levels perform operational control of the system. Besides, at any level, the controllers communicate their decisions to the lower levels or even negotiate their decisions with the higher levels.

Parameterized MPC [105] is based on a receding-horizon control scheme and parameterized control laws. More specifically, in parameterized MPC, the control inputs are parameterized and only the parameters are optimized with respect to the system performance. Generally, the number of parameters is much smaller than the number of control inputs over the control horizon. Therefore, the computational load for parameterized MPC is much less than the standard MPC.

In the rest of this thesis we mainly apply the aforementioned alternative MPC methods to the two main research topics: control of urban transportation networks and control of large-scale hybrid systems.

2.2 Traffic management

Traffic management and control consists in obtaining traffic information, applying traffic control, managing traffic demands and incident, etc. Its goal is to provide safe, reliable, and sustainable travel in a changing environment, and at the same time, to take into account economical, social and environmental factors, such as the total time spent and total energy consumption of all vehicles. In this section we present an overview on traffic management

and control by introducing some traffic flow models, traffic energy consumption models, traffic management measures, and traffic control design methods for traffic networks.

2.2.1 Traffic flow models

Depending on the level of accuracy in describing the dynamical behaviors of individual vehicles, the traffic flow models are typically classified into the following two categories:

- Microscopic traffic flow model
- Macroscopic traffic flow model

For the sake of simplicity, we only briefly describe the general concepts of the traffic flow models in this section.

Microscopic traffic flow models

Microscopic traffic flow models aim to describe the dynamics of individual vehicles [8, 125], including speeding up, braking, cruising, lane changing, and maintaining a safe distance from other vehicles, etc. Therefore, microscopic traffic flow models can simulate the dynamics of traffic flows in a detailed way and thus are often used as simulators for evaluating the effectiveness of traffic control approaches. There have been some traffic flow simulators developed based on microscopic traffic flow models, e.g. VISSIM developed by PTV Group, Germany, and SUMO developed by the German Aerospace Center, Germany. However, due to the high computational complexity, microscopic traffic flow models are generally not used as prediction models in model-based control approaches. For this reason, macroscopic traffic flow models, where the dynamics of traffic flows are described in an aggregated way, are often used for on-line model-based control of large-scale traffic networks.

Macroscopic traffic flow models

In macroscopic traffic flow models [8, 125], traffic flows are often considered to be similar to fluid flows, and the dynamics of traffic flows are described through aggregated traffic variables. Generally, in macroscopic traffic flow models, the dynamics of traffic flows over different locations and different time periods are typically captured by the following three aggregated variables:

- **"Mean speed"** is defined as the average speed of vehicles over a time period (time-mean speed) or over an area (space-mean speed).
- **"Density"** is defined as the number of vehicles per unit length of the roadway.
- **"Flow"** is defined as the number of vehicles passing a reference point of the roadway per unit of time.

Actually, a fundamental diagram is a mathematical description of the approximate equilibrium relationship among these three aggregated variables [51, 66]. More specifically, if one of the three aggregated variables is given, the other two can be determined using a fundamental diagram. So far, most of the macroscopic traffic flow models, Lighthill-Whitham-Richards (LWR) model [109], Cell Transmission Model (CTM) [32, 33], and METANET model [92] etc, have been developed based on the fundamental diagram.

2.2.2 Traffic energy consumption models

Traffic energy consumption models describe the energy consumption of traffic flows based on their dynamics. Since the models of the dynamics of traffic flows are classified into the microscopic category and the macroscopic category, the models of the energy consumption of vehicles are also classified into those two categories accordingly.

Microscopic traffic energy consumption models

Microscopic traffic energy consumption models aim to describe the energy consumption of individual vehicles over time and space. There have been several microscopic traffic energy consumption models, e.g., CMEM [5], COPERT [134] and VERSIT+ [74], developed in the literature. Due to the high accuracy in modeling the energy consumption in traffic networks and the high computational complexity, microscopic traffic energy consumption models are often used to evaluate the performance of traffic control approaches instead of being used for on-line control.

Macroscopic traffic energy consumption models

Macroscopic traffic energy consumption models describe the energy consumption in traffic networks based on aggregating the energy consumption factors of vehicles over time and space. There have been several macroscopic traffic energy consumption models, MOBILE6 [133] and VT-macro [135], developed in the literature. Like macroscopic traffic flow models, macroscopic traffic energy consumption model are mainly used as the prediction model for on-line control purpose.

2.2.3 Traffic management measures

In order to manage a large-scale traffic network, many decision-making processes have to be involved. A general description of the decision-making processes involved in the management of transportation networks is as follows:

- **Strategic decisions** are the long-term decisions related to the construction and changes of the transportation networks, e.g., building new road sections.
- **Tactical decisions** are the mid-term decisions related to the effective use of the existing transportation networks, e.g., determining the orientation of road sections [93].
- **Operational decisions** are the short-term decisions related to scheduling or traffic flow control, e.g., scheduling the traffic lights [37], determining speed limits on the roadways [57], and routing the traffic flows [30], etc.

Generally, expanding the existing infrastructures of a transportation network is the most straightforward way to deal with congestion. However, it is not the most efficient one. First of all, it is monetarily costly and time consuming. Second, it may cause temporary blockage in the network because of ongoing construction. Third, it is often only a temporary solution and might not be able to satisfy the future demand. On the contrary, the tactical and the operational management, which focus on making effective use of existing infrastructures, are more sustainable and promising measures to tackle traffic congestion. From the control

point of view, most existing literature has focused on the operational management of the transportation networks.

2.2.4 Control design methods

In the literature there have been many different control design methods developed for managing and controlling large-scale traffic networks [34, 99, 135]. In this section, we will discuss the control design methods that are often used in practice such as

- Optimal control
- Model predictive control
- Parameterized feedback control
- Artificial-intelligence-based control

Note that among these control design methods, we have already presented the first two in Section 2.1. Hence, in this section we only discuss the other two.

Parameterized feedback control

In order to deal with modeling errors and disturbances, feedback based on the state or the output of a system is always involved in the control loop of the system. In parameterized feedback control, the feedback control law is parameterized and the parameters need to be tuned before the control law takes effect. Depending on whether the parameters are tuned on-line, feedback control can be divided into static feedback control and dynamic feedback control. In static feedback control methods, e.g., PID control, the parameters of the control law are tuned off-line and then are taken to be fixed for on-line control [6]. In dynamic feedback control methods, the parameters of the control law are updated via on-line optimization at every control step [135].

Artificial-intelligence-based control

Artificial intelligence techniques have been used in controller design for large-scale systems. In artificial-intelligence-based control methods, artificial intelligence techniques enable the controllers to reason about the problems and to take actions accordingly [127]. The artificial-intelligence-based control methods that are often used in controller design are:

- **Fuzzy control** determines the control inputs of a system based on a decision-making process consisting of fuzzification, rule-based inference, and defuzzification [41].
- **Neural-network-based control** determines the control inputs of a system by trying to mimic the way in which the human brain processes information [75].
- **Reinforcement learning control** determines the control inputs of a system by enabling the controller to learn [2].
- Other artificial-intelligence-based control methods include those based on swarm intelligence [17], case-based reasoning [1], and Bayesian networks [61], and those based on combining different artificial intelligence techniques, such as fuzzy neural networks [60].

Since the artificial-intelligence-based control methods are not used in this thesis, we will not explicitly discuss them. Interested readers are suggested to refer to the references mentioned above.

2.3 Control of hybrid systems

A hybrid system is a dynamic system that contains both continuous components and discrete components. Therefore, a hybrid system can exhibit continuous behavior, e.g., motion, and discrete behavior, e.g., mode switches. In this section we present an introduction to control of hybrid systems, including description of the modeling frameworks and of the control design methods.

2.3.1 Modeling frameworks

Models are the ultimate tools for understanding the complex interaction between the continuous dynamics and the discrete dynamics of hybrid systems. Some well-known classes of modeling frameworks that are often used to describe hybrid systems are:

- Mixed logical dynamic models [11]
- Piecewise affine models [117]
- Max-min-plus-scaling models [38]
- Linear complementarity systems [55]
- Extended linear complementarity systems [36]
- Switched systems [73]
- Hybrid automata [3]

In this section we briefly illustrate mixed logical dynamic models, piecewise affine models, and max-min-plus-scaling models. For the other models, the readers are suggested to refer to the references given above.

Mixed logical dynamic models

Mixed logical dynamic models describe linear hybrid systems using interdependent physical laws, logic rules and operational constraints [11]. They are given by linear dynamic equations subject to linear inequalities involving real and integer variables:

$$x(k+1) = Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k), \tag{2.3a}$$

$$y(k) = Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k), \tag{2.3b}$$

$$E_1 x(k) + E_2 u(k) + E_3 \delta(k) + E_4 z(k) \leq e_5 \tag{2.3c}$$

where $x(k) = \left[x_{\text{real}}^{\text{T}}(k) \;\; x_{\text{binary}}^{\text{T}}(k)\right]^{\text{T}}$ with $x_{\text{real}}(k)$ denoting the real state variables and $x_{\text{binary}}(k)$ denoting the binary state variables. Similarly, the system output and the control input have the structure $y(k) = \left[y_{\text{real}}^{\text{T}}(k) \;\; y_{\text{binary}}^{\text{T}}(k)\right]^{\text{T}}$ and $u(k) = \left[u_{\text{real}}^{\text{T}}(k) \;\; u_{\text{binary}}^{\text{T}}(k)\right]^{\text{T}}$. Besides, $z(k)$ and $\delta(k)$ are auxiliary real and binary variables, respectively.

Piecewise affine models

Piecewise affine models [117] are extended from linear models and are capable of handling hybrid phenomena. Mathematically, piecewise affine models are given by

$$x(k+1) = A_i x(k) + B_i u(k) \tag{2.4a}$$
$$y(k) = C_i x(k) + D_i u(k) \tag{2.4b}$$
$$\text{for } \begin{bmatrix} x^{\mathrm{T}}(k) & u^{\mathrm{T}}(k) \end{bmatrix}^{\mathrm{T}} \in \Omega_i \tag{2.4c}$$

where $x(k) \in \mathbb{R}^{n_{\mathrm{x}}}$, $u(k) \in \mathbb{R}^{n_{\mathrm{u}}}$, $y(k) \in \mathbb{R}^{n_{\mathrm{y}}}$ denote the state, the control input and the output of the system, respectively, and Ω_i denotes a convex polyhedra in the space $\mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}}$. Note that $\{\Omega_i\}_i^N$ is the partition of $\mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}}$. Theoretically, piecewise affine models are able to model nonlinear and nonsmooth processes with arbitrary accuracy.

Max-min-plus-scaling models

Max-min-plus-scaling expressions are built using operations including maximization, minimization, addition and scalar multiplication and are defined by the following recursive grammar

$$f := \alpha f_1 \big| f_2 + \beta \big| x_i \big| f_2 \big| \min(f_1, f_2) \big| \max(f_1, f_2) \tag{2.5}$$

where the symbol $\big|$ stands for OR, x_i with $i \in \{1, 2, ..., n\}$ denote the variables, $\alpha, \beta \in \mathbb{R}$ are constants, and f_1, f_2 are also max-min-plus-scaling expressions. More specifically, an example of a max-min-plus-scaling expression is $7x_3 + \max\big(4x_1 + x_2, \min(2x_1 + 5x_2, 3x_3)\big)$.

Max-min-plus-scaling models have been introduced to describe discrete event systems [38]. Mathematically, they are given by

$$x(k+1) = f_{\mathrm{s}}\big(x(k), u(k), z(k)\big) \tag{2.6a}$$
$$y(k) = f_{\mathrm{o}}\big(x(k), u(k), z(k)\big) \tag{2.6b}$$
$$f_{\mathrm{c}}\big(x(k), u(k), z(k)\big) \leq c \tag{2.6c}$$

where all elements of $f_{\mathrm{s}}, f_{\mathrm{o}}$ and f_{c} are max-min-plus-scaling expressions in terms of the state variables $x(k)$, the control input $u(k)$ and the auxiliary variables $z(k)$.

Equivalence of different modeling frameworks

Each class of modeling framework has its own advantages in describing hybrid systems that belong to this class. More specifically, there have been control and verification techniques proposed for mixed logical dynamic models [11, 12], stability criteria for piecewise affine models [62], and conditions of well-posedness for linear complementarity systems [55]. Therefore for the study of a particular hybrid system, it would be beneficial to choose the most appropriate modeling framework. Actually, mixed logical dynamic models, piecewise affine models and max-min-plus-scaling models can be equivalently transformed into each other [56]. This is of great importance in the sense that it facilitates the transfer of theoretical results and synthesis tools from one class to any of the equivalent classes.

2.3.2 Control design methods

There have been many methods presented in the literature for the control of hybrid systems, such as optimal control [14, 67], model predictive control [22, 83], game-theoretic control [9, 124], and sliding mode control [114, 131]. In this thesis, we focus on model predictive control of a class of hybrid systems with limited information sharing. Note that the theoretical framework of model predictive control have been described in Section 2.1.2. For the sake of simplicity, we will not discuss the other control approaches, but we refer the interested readers to the references given above.

The major challenge

The optimization-based control methods, such as optimal control and model predictive control, used for control of hybrid systems were originally developed for control of finite-dimensional continuous-time systems or discrete-time systems. However, due to the interaction between the continuous dynamics and the discrete dynamics, it is quite challenging to employ these methods for control of hybrid systems. The major challenge lies in the fact that the resulting optimization problems that need to be solved to determine the control inputs are computationally very complex. More specifically, due to the existence of discrete variables, the resulting optimization problems for determining the control inputs are actually *integer programming* problems or *mixed integer programming* problems. Generally, these problems are computationally very hard to solve, especially when the number of integer variables is large.

2.4 Summary

In this chapter the background of this thesis was explained. First, we have presented the theoretical frameworks of optimal control and model predictive control, which will be used later on in this thesis for both control of urban traffic networks and control of hybrid systems. Since centralized MPC may become intractable in practice for large-scale systems, we have also discussed some alternative MPC approaches including decentralized MPC, distributed MPC, hierarchical MPC and parameterized MPC. After that, we have presented an overview on traffic management and control by introducing traffic flow models, traffic energy consumption models, traffic management measures, and traffic control design methods. Finally, we have presented an introduction to control of hybrid systems including a description of the modeling frameworks and of control design methods.

Part I

Urban traffic routing

Chapter 3

Modeling of the Dynamics and the Energy Consumption of Cybercars

Automated driving technologies have already been developed for individual vehicles and for platoons of vehicles. However, the lack of efficient methods for the control of all vehicles in a road network is one of the biggest challenges that cybercars (i.e., fully automatic road vehicles providing on-demand and door-to-door transportation service) are facing. When it comes to fleet control, where all vehicles in a network is considered as a whole fleet, cybercars can be characterized as moving decision making agents with on-board processing and communication capabilities as well as abundant information of the environment. Before an efficient fleet control method can be developed, a reasonably accurate and sufficiently fast model of the dynamics of cybercars that is suited for control design is needed. In this chapter, we present the modeling of the dynamics and the energy consumption of cybercars in a cybernetic transportation network. The chapter is organized as follows. In Section 3.1, we present the general description of a cybernetic transportation network and the assumptions we make for modeling the dynamics and the energy consumption of cybercars. Afterwards, in Section 3.2, we model the dynamics and the energy consumption of cybercars using a discrete-time modeling method. Next, in Section 3.3, we present a discrete-event model of the dynamics and the energy consumption of cybercars. Finally, in Section 3.4, we briefly discuss the features of the two modeling methods and summarize this chapter.

Parts of this chapter have been published in [76] and [81].

3.1 General description

We consider a cybernetic transportation network (i.e. only open to cybercars) consisting of a set of roads and a set of intersections. For the sake of simplicity, we refer to an intersection as a 'node', and a road section between two intersections as a 'link'. Each link is divided into a number of segments that have typical lengths ranging between 50 m and 100 m. We assume that, at any time, the desired speed of all cybercars in a segment is determined by the traffic density (i.e., the number of vehicles per kilometer) in that segment, while the actual speed of each cybercar is determined by its previous speed and its current desired speed. Moreover, we assume each segment has a maximum allowed number of cybercars at the same time (i.e. maximum capacity). More specifically, if the number of cybercars in a segment reaches or exceeds the maximum capacity, that segment will be blocked. A blocked segment will be

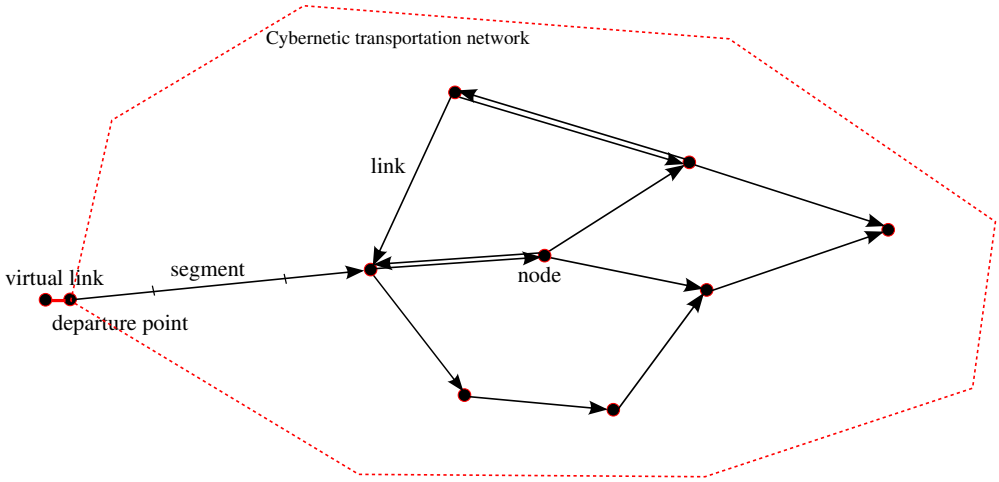

Figure 3.1: Cybernetic transportation network

unblocked immediately when the number of cybercars in that segment becomes lower than the maximum capacity. At any time, the energy consumption of a cybercar is a function of its velocity, its acceleration (or deceleration), as well as its position (related to its potential-energy change in case of link with slope). Without loss of generality, we assume the origins and the destinations of all cybercars are always at nodes. Besides, we also assume that no cybercar can cover a distance longer than the length of a segment within one simulation time interval. Finally, we assume there are higher-level controllers assigning transport service requests (including starting time, origin and destination) to each cybercar and we only focus on modeling the dynamics and the energy consumption of cybercars.

In case of blocked departure links, cybercars are not able to enter the network even when the times at which they are due to depart have come. In order to model the queues of cybercars waiting at the origins due to blocked departure links, to each departure point, we introduce a virtual link with zero length and infinite capacity. Without loss of generality, the layout of a cybernetic transportation network can then conceptually be represented by the graph shown in Figure 3.1, where a node is represented by a small solid circle while a link is represented by a directed line with the arrow indicating the heading direction.

3.2 Discrete-time model

In this section, we present a discrete-time model of the dynamics and the energy consumption of cybercars.

3.2.1 Definitions

Let T denote the length of the simulation time interval and let k denote the discrete-time step counter. Let $T_{\text{start},i}$ denote the starting time of cybercar i and $T_{\text{stop},i}$ denote the arrival time of cybercar i at its destination. Let $v_i(k)$ denote the speed of cybercar i at time kT, $l_i(k)$ and $s_i(k)$ respectively denote the link and the segment in which cybercar i is running at time kT, and $x_i(k)$ denote the position (measured along the longitudinal axis of a link) of cybercar i in $l_i(k)$ at time kT. Let $l_{\text{final},i}$ denote the final link of cybercar i (i.e., the end of $l_{\text{final},i}$ is the destination of cybercar i). Let $r_i(k)$ denote the selected route of cybercar i

at time kT, and $u_i(k)$ denote the next link of cybercar i after $l_i(k)$ on route $r_i(k)$. Let $p_{m,j}^{\text{start}}$ and $p_{m,j}^{\text{end}}$ respectively denote the positions (measured along longitudinal axis) of the starting point and the end point of segment m of link j, and let $v_{\text{free},m,j}$ denote the free-flow speed[1] for segment m of link j. Letting $N_{m,j}(k)$ be the number of cybercars in segment m of link j at time kT, the traffic density in the segment is given by

$$\rho_{m,j}(k) = \frac{N_{m,j}(k)}{L_{m,j}} \tag{3.1}$$

where $L_{m,j}$ denotes the length of segment m of link j. Besides, let $C_{m,j}(k)$ denote the maximum capacity of segment m of link j and let $b_{m,j}(k)$ denote the binary blocking signal of the segment with $b_{m,j}(k) = 1$ the segment is blocked. Finally, let $\Delta_{m,j}$ denote the change of the number of cybercars in segment m of link j during one simulation time interval. Note that we set $\Delta_{m,j}$ for all m and j to be 0 at the start of every simulation time interval.

3.2.2 Equilibrium speed-flow relationship considering slope

Vehicles tend to accelerate when going downhill while they tend to decelerate when going uphill. Based on the experimental results of geometric effects on the speeds of vehicles presented in [132], the impact of the slope on the free-flow speed of a segment can be modeled as follows:

$$v_{\text{free},m,j} = \begin{cases} v_{\text{free},m,j,0}\Big(1 - \delta_{\text{up},m,j}\tan(\vartheta_{m,j})\Big), & \text{if } \vartheta_{m,j} \geq 0 \\ v_{\text{free},m,j,0}\Big(1 + \delta_{\text{down},m,j}\tan(\vartheta_{m,j})\Big), & \text{if } \vartheta_{m,j} < 0 \end{cases}$$

where $\vartheta_{m,j}$ denotes the angle of inclination of segment m of link j, $v_{\text{free},m,j,0}$ denotes the free-flow speed of vehicles in segment m of link j if that segment is flat, $\delta_{\text{up},m,j}$ and $\delta_{\text{down},m,j}$ are relative terms that denote the impact of each 1% downhill and uphill grade on the free-flow speed of cybercars in segment m of link j, respectively.

Besides, the critical traffic density of segment m of link j at which the maximal flow is obtained may also be influenced by the slope. Also inspired by [132], one possible way to model this influence is given by

$$\rho_{\text{crit},m,j} = \rho_{\text{crit},m,j,0}\Big(1 + \alpha_{m,j}\tan(\vartheta_{m,j})\Big)$$

where

$$\rho_{\text{crit},m,j,0} = \frac{1}{h_{\text{con},m,j}\, v_{\text{free},m,j,0} + L_{\text{veh}}}$$

is the critical traffic density of segment m of link j if that segment would be flat, $h_{\text{con},m,j}$ is the constant time headway of automated vehicles on segment m of link j, L_{veh} is the average length of vehicles, and $\alpha_{m,j}$ is a model parameter.

Finally, according to the macroscopic characteristics of semi-automated traffic presented in [18], the equilibrium speed-flow relationship of cybercars in a segment with slope is given

[1] Free-flow speed is the speed that the drivers would drive when the traffic density on the road is very low and the average distance headway is large.

by

$$V_{m,j}\big(\rho_{m,j}(k)\big) = \begin{cases} v_{\text{free},m,j}, & \text{if } \rho_{m,j}(k) \le \rho_{\text{crit},m,j} \\ \frac{c_{m,j}}{\rho_{m,j}(k)} + d_{m,j}, & \text{if } \rho_{m,j}(k) > \rho_{\text{crit},m,j} \end{cases} \tag{3.2}$$

with

$$c_{m,j} = \frac{v_{\text{free},m,j} \cdot \rho_{\text{crit},m,j} \cdot \rho_{\text{jam},m,j}}{\rho_{\text{jam},m,j} - \rho_{\text{crit},m,j}}$$

$$d_{m,j} = -\frac{v_{\text{free},m,j} \cdot \rho_{\text{crit},m,j}}{\rho_{\text{jam},m,j} - \rho_{\text{crit},m,j}}$$

where $\rho_{\text{jam},m,j}$ is the jam traffic density of segment m of link j, i.e. the density at which the traffic flow is 0 veh/h.

3.2.3 Speed change of a single cybercar

In the discrete-time modeling framework, the speed of a cybercar is assumed to be fixed within one simulation time interval and is updated only at the end of the simulation time interval. In particular, the update of the speed of cybercar i at simulation time step k is given by

$$v_i(k+1) = v_i(k) + \xi_i\Big(v_{\text{desired},i}(k) - v_i(k)\Big) \tag{3.3}$$

where $v_{\text{desired},i}(k)$ denotes the desired speed of cybercar i at kT and it is determined by (3.2), and ξ_i indicates how fast cybercar i can change its speed based on the difference between its desired speed and its current speed. To be more specific, ξ_i is given by

$$\xi_i = \frac{a_{\text{max},i}\,T}{v_{\text{max},i}} < 1 \tag{3.4}$$

where $v_{\text{max},i}$ and $a_{\text{max},i}$ are the maximal speed and maximal acceleration rate of cybercar i, respectively.

3.2.4 Dynamics of a single cybercar

Each cybercar i enters the network at $T_{\text{start},i}$. After that, at each simulation time step kT, with $x_i(k)$, $v_i(k)$, $l_i(k)$, $s_i(k)$, $r_i(k)$ and $N_{m,j}(k)$, $\rho_{m,j}(k)$, $b_{m,j}(k)$ for all j and m given, the variables $x_i(k+1)$, $v_i(k+1)$, $l_i(k+1)$, and $s_i(k+1)$ of cybercar i need to be determined. As cybercar i may go from one segment (or link) to a different segment (or link) during one simulation time interval, the change of the number of vehicles in the segments due to the change of the position of cybercar i also needs to be captured.

From simulation time step kT to step $(k+1)T$, the update of the dynamics of a single cybercar i can be divided into five cases, which are characterized as follows:

- "**same segment, same link**": cybercar i stays in the same segment and the same link.
- "**different segments, same link**": cybercar i goes from one segment to the next one in the same link.

- "**desired segment blocked**": cybercar i reaches the end of its current segment, but its desired next segment is blocked.

- "**different links**": cybercar i goes from its current link to its desired next link.

- "**arrival**": cybercar i arrives at its destination.

For the sake of simplicity of notation, in the following, we assume $l_i(k) = j$ and $s_i(k) = m$ when we describe the update of the dynamics of cybercar i in each of the cases.

First, the conditions for the case of **same segment, same link** are:

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \left[v_i(k) + \xi_i \left(V_{m,j}\left(\rho_{m,j}(k)\right) - v_i(k) \right) \right] T \leq p_{m,j}^{\text{end}}$$

where the function $V_{m,j}(\cdot)$ describes how the equilibrium speed of cybercars in segment m of link j depends on the traffic density in that segment. One possible way to define $V_{m,j}(\cdot)$ has been given in the Section 3.2.2. The dynamics of cybercar i are then updated by

$$v_i(k+1) = v_i(k) + \xi_i \left(V_{m,j}\left(\rho_{m,j}(k)\right) - v_i(k) \right)$$
$$x_i(k+1) = x_i(k) + v_i(k+1)T$$
$$l_i(k+1) = l_i(k)$$
$$s_i(k+1) = s_i(k)$$

For the case of **different segments, same link**, the following conditions must be satisfied:

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \left[v_i(k) + \xi_i \left(V_{m,j}\left(\rho_{m,j}(k)\right) - v_i(k) \right) \right] T > p_{m,j}^{\text{end}}$$
$$b_{m+1,j}(k) = 0$$

In this case, cybercar i first runs at the speed $v_{\text{aux},i}(k) = v_i(k) + \xi_i \left(V_{m,j}\left(\rho_{m,j}(k)\right) - v_i(k) \right)$ in the current segment. After reaching the end of the current segment, it enters the next segment and runs at $v_{\text{aux},i}(k) + \xi_i \left(V_{m+1,j}\left(\rho_{m+1,j}(k)\right) - v_{\text{aux},i}(k) \right)$ for the rest of the time interval. Then the dynamics of cybercar i are updated by

$$v_i(k+1) = v_{\text{aux},i}(k) + \xi_i \left(V_{m+1,j}\left(\rho_{m+1,j}(k)\right) - v_{\text{aux},i}(k) \right)$$
$$x_i(k+1) = p_{m,j}^{\text{end}} + v_i(k+1)\tau_i$$
$$l_i(k+1) = l_i(k)$$
$$s_i(k+1) = s_i(k) + 1$$

where τ_i denotes the remaining time during $[kT, (k+1)T]$ after cybercar i arrives at the end of the current segment m:

$$\tau_i = T - \frac{p_{m,j}^{\text{end}} - x_i(k)}{v_{\text{aux},i}(k)}$$

Besides, the changes of the number of cybercars in segment m and segment $m+1$ are captured by

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$
$$\Delta_{m+1,j} \leftarrow \Delta_{m+1,j} + 1$$

Next, the conditions for the case of **desired segment blocked** are given by

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \left[v_i(k) + \xi_i \left(V_{m,j}\left(\rho_{m,j}(k)\right) - v_i(k) \right) \right] T > p^{\text{end}}_{m,j}$$
$$b_{m+1,j}(k) = 1$$

Given the desired next segment is currently blocked, cybercar i has to wait after arriving at the end of the current segment. Therefore, the update of dynamics of cybercar i is given by

$$v_i(k+1) = 0$$
$$x_i(k+1) = p^{\text{end}}_{m,j}$$
$$l_i(k+1) = l_i(k)$$
$$s_i(k+1) = s_i(k)$$

The conditions for the case of **different links** are:

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \left[v_i(k) + \xi_i \left(V_{m,j}\left(\rho_{m,j}(k)\right) - v_i(k) \right) \right] T > p^{\text{end}}_{m,j}$$
$$b_{1,j^*}(k) = 0$$

where j^* denotes the planned next link of cybercar i at kT. In this case, cybercar i enters link j^*, and its dynamics are updated by

$$v_i(k+1) = v_{\text{aux},i}(k) + \xi_i \left(V_{1,j^*}\left(\rho_{1,j^*}(k)\right) - v_{\text{aux},i}(k) \right)$$
$$x_i(k+1) = v_i(k+1)\tau_i$$
$$l_i(k+1) = u_i(k)$$
$$s_i(k+1) = 1$$

The changes of the number of cybercars in the last segment of link j and the first segment of link j^* are then captured by

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$
$$\Delta_{1,j^*} \leftarrow \Delta_{1,j^*} + 1$$

Finally, for the case of **arrival**, the conditions are given by

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \left[v_i(k) + \xi_i \left(V\left(\rho_{m,j}(k)\right) - v_i(k) \right) \right] T \geq p^{\text{end}}_{m,j}$$

$$l_{\text{final},i} = j$$

In this case, cybercar i reaches its destination and its arrival time $T_{\text{stop},i}$ is obtained by

$$T_{\text{stop},i} = kT + \frac{p_{m,j}^{\text{end}} - x_i(k)}{v_{\text{aux},i}(k)} \tag{3.5}$$

We assume cybercar i leaves the network after arriving at its destination. Then $\Delta_{m,j}$ is updated by

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$

3.2.5 Dynamics of the network

At every simulation time step, after the dynamics of all cybercars are updated, the states of the whole network are updated by

$$N_{m,j}(k+1) = N_{m,j}(k) + \Delta_{m,j}$$
$$\rho_{m,j}(k+1) = \frac{N_{m,j}(k+1)}{L_{m,j}}$$
$$b_{m,j}(k+1) = \mathbb{1}\Big(N_{m,j}(k+1) \geq C_{m,j}\Big)$$

where $\mathbb{1}(\cdot)$ is an indicator function defined by

$$\mathbb{1}(a) = \begin{cases} 1, & \text{if } a \text{ is true} \\ 0, & \text{else} \end{cases}$$

3.2.6 Energy consumption of a single cybercar

Generally, the energy consumption of a cybercar consists of the following five factors [84]:

- speeding up
- air drag
- rolling resistance
- going uphill
- energy losses in the energy-conversion chain

Assuming each cybercar has the same acceleration and deceleration rate, we use the graphs in Figure 3.2 to calculate the first three consuming factors of a cybercar. For notational convenience, in this section, we drop the subscript indicating the index of the cybercar for all the variables. Note that in Figure 3.2, v_{init} and v_{new} respectively denote the speed of the cybercar at the beginning and the speed at the end of a simulation interval, and t_1 denotes the absolute time instant at which v_{new} is reached. According to (3.3) and (3.4), $t_1 < (k+1)T$ always holds.

It should also be noted that the effects of acceleration and deceleration of a cybercar are non-negligible in the calculation of its energy consumption. Therefore, different from

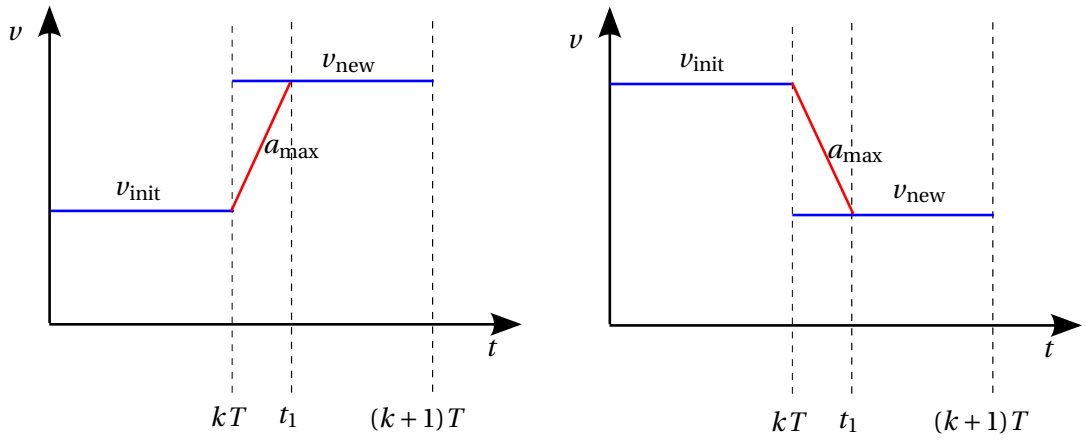

Figure 3.2: Two possible cases of the speed change of a cybercar

Section 3.2.4 where the dynamics of a cybercar are updated assuming a constant speed in every simulation interval, in this subsection, the acceleration and deceleration processes of a cybercar are approximated and then taken into account in the energy consumption calculation. In fact, acceleration and deceleration could also have been taken into account in the update of the dynamics of a cybercar in Section 3.2.4, but the relative effect of that would be much smaller than for the energy consumption calculation.

According to Figure 3.2, the cybercar is first accelerating or decelerating at a rate $a_{\max}$ during $[kT, t_1]$, which we define as the acceleration-deceleration period. For the cybercar in this period, let $E_{\text{var\_kin}}$ denote the change of kinetic energy, $E_{\text{var\_air}}$ the energy consumption needed to overcome the air drag, and $E_{\text{var\_rol}}$ the energy consumption needed to overcome the rolling resistance. Based on [84], we have

$$E_{\text{var\_kin}} = M \int_{v_{\text{init}}}^{v_{\text{new}}} v\,dv = \frac{1}{2} M (v_{\text{new}}^2 - v_{\text{init}}^2) \tag{3.6}$$

$$E_{\text{var\_air}} = \int_{kT}^{t_1} \frac{1}{2} \rho_{\text{air}} A_{\text{front}} v^3\,dt = \frac{\rho_{\text{air}} A_{\text{front}} \left| v_{\text{new}}^4 - v_{\text{init}}^4 \right|}{8 a_{\max}} \tag{3.7}$$

$$E_{\text{var\_rol}} = \int_{kT}^{t_1} c_{\text{r}} M g v\,dt = \frac{c_{\text{r}} M g \left| v_{\text{new}}^2 - v_{\text{init}}^2 \right|}{2 a_{\max}} \tag{3.8}$$

where M and A_{front} denote respectively the mass and the effective frontal area of the cybercar, ρ_{air} denotes the air density, c_{r} denotes the rolling resistance coefficient, and g denotes the gravitational acceleration.

After the acceleration-deceleration period, the cybercar keeps a constant speed v_{new} for the rest of the simulation interval $[t_1, (k+1)T]$, which we define as the constant-speed period. In this period, the kinetic energy of the cybercar does not change. Then by defining $E_{\text{ct\_air}}$ and $E_{\text{ct\_rol}}$ respectively as the energy consumption of the cybercar to overcome the air drag and the rolling resistance during the constant-speed period, we have

$$E_{\text{ct\_air}} = \frac{1}{2} \rho_{\text{air}} A_{\text{front}} v_{\text{new}}^3 \cdot \left(T - \frac{|v_{\text{new}} - v_{\text{init}}|}{a_{\max}} \right) \tag{3.9}$$

$$E_{\text{ct\_rol}} = c_{\text{r}} M g v_{\text{new}} \cdot \left(T - \frac{|v_{\text{new}} - v_{\text{init}}|}{a_{\max}} \right) \tag{3.10}$$

Furthermore, we consider the change of the potential energy of the cybercar, which is denoted by E_{pot}, during one simulation interval. If the cybercar stays in the same segment m of the same link j within the simulation interval, given the angle of inclination $\vartheta_{m,j}$ of the segment, we have

$$E_{\text{pot}} = Mg \sin(\vartheta_{m,j}) \cdot (x_{\text{new}} - x_{\text{init}}) \tag{3.11}$$

where x_{new} and x_{init} are the positions of the cybercar in the link at the beginning and the end of the simulation interval, respectively. If the cybercar goes from one segment to another within the simulation interval, we have

$$E_{\text{pot}} = Mg \left[\sin(\vartheta_{m,j})(p_{m,j}^{\text{end}} - x_{\text{init}}) + \sin(\vartheta_{m+1,j})(x_{\text{new}} - p_{m,j}^{\text{end}}) \right] \tag{3.12}$$

Moreover, if the cybercar goes from the last segment of the current link to the first segment of the next link within the simulation interval, we get

$$E_{\text{pot}} = Mg \left[\sin(\vartheta_{m,j})(p_{m,j}^{\text{end}} - x_{\text{init}}) + \sin(\vartheta_{1,j^*}) x_{\text{new}} \right] \tag{3.13}$$

By defining η_{motor} as the efficiency of electric motors[2], the actual energy consumption of the cybercar during the simulation time interval $[kT, (k+1)T]$ becomes

$$E(v_{\text{init}}, v_{\text{new}}, x_{\text{init}}, x_{\text{new}}, \vartheta_{m,j}, \vartheta_{m+1,j}, a_{\max}, T) = \max\left(\frac{E_{\text{total}}}{\eta_{\text{motor}}}, 0 \right) \tag{3.14}$$

where

$$E_{\text{total}} = E_{\text{var\_kin}} + E_{\text{var\_air}} + E_{\text{var\_rol}} + E_{\text{ct\_air}} + E_{\text{ct\_rol}} + E_{\text{pot}} \tag{3.15}$$

Moreover, if a cybercar uses regenerative braking, it could save part of the energy lost in braking to recharge its onboard battery. By letting γ_{recover} denote the round-trip energy recovery coefficient[3] of the regenerative braking system, we have

$$E(v_{\text{init}}, v_{\text{new}}, x_{\text{init}}, x_{\text{new}}, \vartheta_{m,j}, \vartheta_{m+1,j}, a_{\max}, T) = \begin{cases} \dfrac{E_{\text{total}}}{\eta_{\text{motor}}}, & \text{if } E_{\text{total}} \geq 0 \\ \dfrac{\gamma_{\text{recover}} \cdot E_{\text{total}}}{\eta_{\text{motor}}}, & \text{if } E_{\text{total}} < 0 \end{cases} \tag{3.16}$$

[2] According to [84], the maximal efficiency of electric motors is about 85% to 90%. That means for the best case, only 90% of the electricity consumed in charging the onboard battery can be used to power the electric vehicle.

[3] The round-trip energy recovery coefficient (i.e., the ratio between the amount of electric energy recovered from braking and the amount consumed in accelerating) of an electric vehicle is around 38% [106].

Finally, by defining $E_i(k)$ as the amount of energy consumed by cybercar i during $[kT,(k+1)T]$, we have

$$E_i(k) = E\Big(v_i(k), v_i(k+1), x_i(k), x_i(k+1), \vartheta_{s_i(k),l_i(k)}, \vartheta_{s_i(k+1),l_i(k+1)}, a_{\max,i}, T\Big) \tag{3.17}$$

3.3 Discrete-event model

A discrete-event model describes the dynamics of a system over a discrete sequence of events in time. Different from a discrete-time model where the update of the state of a system is be done periodically at every time step, in a discrete-event model the update of the state of a system directly jump in time from one event to the next. In this section, we present a discrete-event model of the dynamics and the energy consumption of cybercars.

3.3.1 Definitions

We define two types of events, namely segment events and cybercar events. More specifically, a segment event is defined as the change of the number of cybercars in a segment. There are three types of cybercar events:

- a cybercar is due to depart
- a cybercar arrives at the end of one segment
- the speed of a cybercar changes due to the occurrence of a segment event

Next, we define n_i as the event counter for cybercar i and define $t_i(n_i)$ as the occurrence time of the n_i-th event for cybercar i. Besides, we define $t_{\text{next},i}$ as the predicted occurrence time of the next event after the latest event for cybercar i. Finally, we define $n_{m,j}$ as the event counter for segment m of link j. Note that for the rest of Section 3.3, the event-based variables are defined in the same way as the corresponding time-based variables in Section 3.2.

3.3.2 Model of the dynamics of cybercars

The discrete-event modeling approach of the dynamics of cybercars consists of the following steps:

- Initialization:
  - for all cybercar i, we have $n_i = 0$, $t_i(n_i) = 0$, $x_i(n_i) = 0$, $v_i(n_i) = 0$, $l_i(n_i) = 0$, $s_i(n_i) = 0$, and $t_{\text{next},i} = T_{\text{start},i}$.
  - for all segment m and all link j, we have $n_{m,j} = 0$, $N_{m,j}(n_{m,j}) = 0$, $b_{m,j}(n_{m,j}) = 0$, and $\rho_{m,j}(n_{m,j}) = \frac{N_{m,j}(n_{m,j})}{L_{m,j}}$
- Step 1: Determine the cybercar corresponding to the earliest next cybercar event:

$$i^* = \arg\min_i t_{\text{next},i}$$

- Step 2: Collect all the cybercars the next event of which will occur in the time window $[t_{\text{next},i^*}, t_{\text{next},i^*} + \tau_{\text{window}}]$ in a set $\mathbb{A}$:

$$\mathbb{A} = \{i | t_{\text{next},i} \leq t_{\text{next},i^*} + \tau_{\text{window}}\}$$

  where τ_{window} denotes the length of the time window and it is typically selected as $\tau_{\text{window}} > T$. Note that if the next events of some cybercars occur in a very short time window, then considering those next events separately in updating the dynamics of cybercars will require frequent event checking and much computation. Since the dynamics of cybercars will not change very much in a very short time, we consider those events collectively to reduce the computational effort in updating the dynamics of cybercars.

- Step 3: For all $i \in \mathbb{A}$, check:

  – For each cybercar i that is due to depart (assuming $u_i(n_i + 1) = j^*$ for the sake of compactness of notation), if $b_{1,j^*}(n_{1,j^*}) = 0$, cybercar i is allowed to enter the network. Then we have

$$\Delta_{1,j^*} \leftarrow \Delta_{1,j^*} + 1$$

  – For each cybercar i that wants to go from one segment to the next one in the same link (assuming $l_i(n_i) = j$ and $s_i(n_i) = m$), if $b_{m+1,j}(n_{m,j}) = 0$, cybercar i is allowed to go from one segment to the next one in the same link ("**different segments, same link**"). Then we have

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$
$$\Delta_{m+1,j} \leftarrow \Delta_{m+1,j} + 1$$

  – For each cybercar i that wants to go from its current link to its desired next link (assuming $l_i(n_i) = j$, $s_i(n_i) = m$ and $u_i(n_i + 1) = j^*$), if $b_{1,j^*}(n_{1,j^*}) = 0$, cybercar i is allowed to go from its current link to its desired next link ("**different links**"). Then we have

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$
$$\Delta_{1,j^*} \leftarrow \Delta_{1,j^*} + 1$$

  – For each cybercar i that arrives at its destination ("**arrival**") (assuming $l_i(n_i) = j$ and $s_i(n_i) = m$), we have

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$

- Step 4: Update the dynamics of each cybercar $i \in \mathbb{A}$ (assuming $l_i(n_i) = j$, $s_i(n_i) = m$ and $u_i(n_i + 1) = j^*$)

  – For each cybercar $i \in \mathbb{A}$ that arrives at the end of the current segment but cannot enter the next segment ("**desired segment blocked**"), we have

$$t_i(n_i + 1) = t_{\text{next},i}$$

$$
\begin{aligned}
v_i(n_i+1) &= 0\\
x_i(n_i+1) &= p^{\text{end}}_{m,j}\\
l_i(n_i+1) &= l_i(n_i)\\
s_i(n_i+1) &= s_i(n_i)
\end{aligned}
$$

– For each cybercar $i \in \mathbb{A}$ that arrives at the end of the current segment and is allowed to enter the next segment ("**different segments, same link**"), we have

$$
\begin{aligned}
t_i(n_i+1) &= t_{\text{next},i}\\
v_i(n_i+1) &= F_{\text{speed}}\Big(v_i(n_i),\ V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big),\ t_i(n_i+1)-t_i(n_i)\Big)\\
x_i(n_i+1) &= p^{\text{end}}_{m,j}\\
l_i(n_i+1) &= l_i(n_i)\\
s_i(n_i+1) &= s_i(n_i)+1
\end{aligned}
$$

where the expression of the functions $F_{\text{speed}}(\cdot)$ is given in Section 3.3.4.

– For each cybercar $i \in \mathbb{A}$ that arrives at the end of its current link and is allowed to enter the first segment of its desired next link ("**different links**"), we have

$$
\begin{aligned}
t_i(n_i+1) &= t_{\text{next},i}\\
v_i(n_i+1) &= F_{\text{speed}}\Big(v_i(n_i),\ V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big),\ t_i(n_i+1)-t_i(n_i)\Big)\\
x_i(n_i+1) &= 0\\
l_i(n_i+1) &= u_i(n_i+1)\\
s_i(n_i+1) &= 1
\end{aligned}
$$

– For each cybercar $i \in \mathbb{A}$ that arrives at its destination ("**arrival**"), we have

$$T_{\text{stop},i} = t_{\text{next},i}$$

- Step 5: After the update of the dynamics for all cybercars $i \in \mathbb{A}$, check for all m and j:

– If $\Delta_{m,j} \neq 0$, then we have

$$
\begin{aligned}
N_{m,j}(n_{m,j}+1) &= N_{m,j}(n_{m,j}) + \Delta_{m,j}\\
\rho_{m,j}(n_{m,j}+1) &= \frac{N_{m,j}(n_{m,j}+1)}{L_{m,j}}\\
b_{m,j}(n_{m,j}+1) &= \mathbb{1}\Big(N_{m,j}(n_{m,j}+1) \geq C_{m,j}\Big)\\
n_{m,j} &\leftarrow n_{m,j}+1
\end{aligned}
$$

- Step 6: Since the number of cybercars in each segment is updated at the end of the time window $[t_{\text{next},i^*}, t_{\text{next},i^*}+\tau_{\text{window}}]$, the cybercars that have not arrived at the end of their current segment at $t = t_{\text{next},i^*}+\tau_{\text{window}}$ have to adapt their speeds afterwards due to the change of the number of cybercars in that segment ("**same segment, same**

link"). We collect those cybercars in a set $\mathbb{B}$.

– Assuming $l_i(n_i) = j$ and $s_i(n_i) = m$, the updated dynamics of each cybercar $i \in \mathbb{B}$ are given by

$$t_i(n_i+1) = t_{\text{next},i^*} + \tau_{\text{window}}$$
$$v_i(n_i+1) = F_{\text{speed}}\Big(v_i(n_i),\ V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big),\ t_i(n_i+1) - t_i(n_i)\Big)$$
$$x_i(n_i+1) = F_{\text{position}}\Big(x_i(n_i),\ v_i(n_i),\ V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big),\ t_i(n_i+1) - t_i(n_i)\Big)$$
$$l_i(n_i+1) = l_i(n_i)$$
$$s_i(n_i+1) = s_i(n_i)$$

where the expression $F_{\text{position}}(\cdot)$ is given in Section 3.3.4.

- Step 7: After updating the dynamics of cybercars and the state of the network, we need to predict the occurrence time of new events (assuming $l_i(n_i+1) = h$ and $s_i(n_i+1) = o$).

– For each cybercar $i \in \mathbb{A}$ that arrives at the end of current segment but cannot enter next segment ("**desired segment blocked**"), its predicted next event is at the end of the time window $[t_{\text{next},i^*},\ t_{\text{next},i^*} + \tau_{\text{window}}]$ when the number of cybercars in each segment is updated. Therefore, we have

$$t_{\text{next},i} \leftarrow t_{\text{next},i^*} + \tau_{\text{window}}$$
$$n_i \leftarrow n_i + 1$$

– For each cybercar $i \in \mathbb{A}$ that arrives at the end of current segment and is allowed to enter the next segment ("**different segments, same link**") or that arrives at the end of current link and is allowed to enter the first segment of its desired next link ("**different links**") is at , we have

$$t_{\text{next},i} \leftarrow F_{t_{\text{next}}}\Big(L_{o,h},\ v_i(n_i+1),\ V_{o,h}\Big(\rho_{o,h}(n_{o,h})\Big),\ t_i(n_i+1)\Big)$$
$$n_i \leftarrow n_i + 1$$

where the expression of the function $F_{t_{\text{next}}}(\cdot)$ is given in Section 3.3.5.

– For each cybercar $i \in \mathbb{B}$ ("**same segment, same link**"), we have

$$t_{\text{next},i} \leftarrow F_{t_{\text{next}}}\Big(p^{\text{end}}_{o,h} - x_i(n_i+1),\ v_i(n_i+1),\ V_{m,j}\Big(\rho_{o,h}(n_{o,h})\Big),\ t_i(n_i+1)\Big)$$
$$n_i \leftarrow n_i + 1$$

- Step 8: go back to step 1 until all cybercars arrive at their destinations.

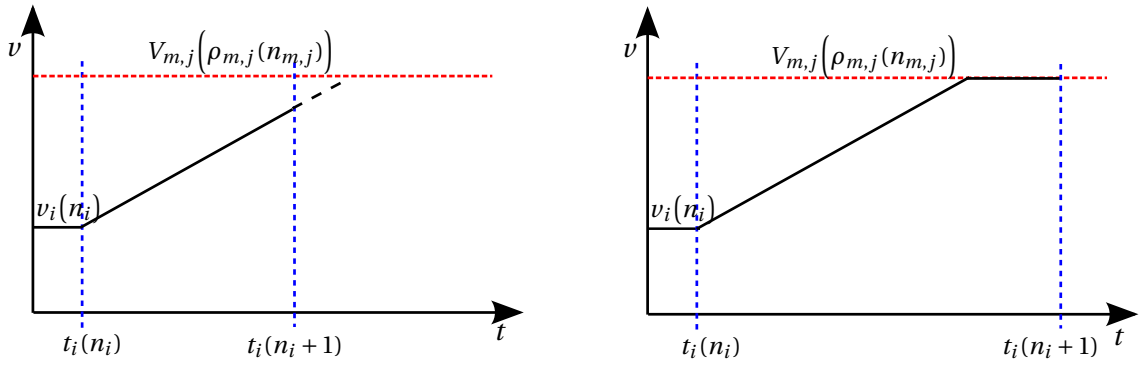

Figure 3.3: Two possible cases of speeding up of a cybercar. In the first case, the cybercar has not reached its desired speed when the next event occurs. In the second case, the cybercar reaches its desired speed before the occurrence of the next event.

3.3.3 Modeling of the energy consumption of cybercars

The energy consumption of each cybercar i between two consecutive events n_i and n_i+1 is given by

$$E_i(n_i) = E\Big(v_i(n_i), v_i(n_i+1), x_i(n_i), x_i(n_i+1), \theta_{s_i(n_i),l_i(n_i)}, \theta_{s_i(n_i+1),l_i(n_i+1)}, a_{\max,i}, t_i(n_i+1)-t_i(n_i)\Big)$$

Note that the expression of $E(\cdot)$ has been given in Section 3.2.6.

3.3.4 Functions to update the speeds and the positions of cybercars

We assume that when a cybercar i needs to adapt its speed due to the occurrence of an event, it accelerates at a constant rate $a_{\text{acc},i}$ or decelerates at a constant rate $a_{\text{dec},i}$ towards its desired speed. Let $a_i(n_i)$ denotes the acceleration or deceleration of cybercar i from its speed at $t_i(n_i)$ towards its desired speed after event n_i. Then $a_i(n_i) = a_{\text{acc},i}$ when cybercar i accelerates and $a_i(n_i) = -a_{\text{dec},i}$ when it decelerates.

There are 2 cases of speeding up of a cybercar depending on whether the time period between the current event n_i and the next event n_i+1 is longer than the time cybercar i needs to reach its desired speed after current event n_i. These two cases are illustrated in Figure 3.3. Based on Figure 3.3, we now explain how the speed and the position of cybercar i are updated at the occurrence time of the next event. Since the cases when cybercar i is accelerating and decelerating are the same except for the vertical position interchange of $v_i(n_i)$ and $V_{m,j}\left(\rho_{m,j}(n_{m,j})\right)$, for the sake of simplicity of demonstration, only the former one is shown in Figure 3.3.

First, if

$$t_i(n_i+1) - t_i(n_i) < \frac{V_{m,j}\left(\rho_{m,j}(n_{m,j})\right) - v_i(n_i)}{a_i(n_i)}$$

then according to the first case shown in Figure 3.3, $v_i\big(n_i+1\big)$ and $x_i(n_i+1)$ are given by

$$v_i\big(n_i+1\big) = v_i\big(n_i\big) + a_i(n_i)\Big[t_i(n_i+1) - t_i(n_i)\Big]$$

$$x_i(n_i+1) = x_i(n_i) + v_i(n_i)\Big[t_i(n_i+1) - t_i(n_i)\Big] + 0.5a_i(n_i)\Big[t_i(n_i+1) - t_i(n_i)\Big]^2$$

Otherwise, according to the second case shown in Figure 1 $v_i(n_i+1)$ and $x_i(n_i+1)$ are given by

$$v_i(n_i+1) = V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big)$$
$$x_i(n_i+1) = x_i(n_i) + d_{\text{var}} + d_{\text{ct}}$$

where d_{var} and d_{ct} denote the distances that the cybercar cover in the acceleration-deceleration period and in the constant-speed period, respectively. More specifically, d_{var} and d_{ct} are determined by

$$d_{\text{var}} = \frac{V_{m,j}^2\Big(\rho_{m,j}(n_{m,j})\Big) - v_i^2(n_i)}{2a_i(n_i)}$$
$$d_{\text{ct}} = V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big) \cdot \left(t_i(n_i+1) - t_i(n_i) - \frac{V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big) - v_i(n_i)}{a_i(n_i)}\right)$$

Therefore, there are two cases:

- case 1:

$$t_i(n_i+1) - t_i(n_i) < \frac{V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big) - v_i(n_i)}{a_i(n_i)}$$

- case 2:

$$t_i(n_i+1) - t_i(n_i) \geq \frac{V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big) - v_i(n_i)}{a_i(n_i)}$$

and $F_{\text{speed}}(\cdot)$ and $F_{\text{position}}(\cdot)$ are given by

$$F_{\text{speed}}(\cdot) = \begin{cases} v_i(n_i) + a_i(n_i)\Big[t_i(n_i+1) - t_i(n_i)\Big], & \text{for case 1} \\ V_{m,j}\Big(\rho_{m,j}(n_{m,j})\Big), & \text{for case 2} \end{cases}$$

$$F_{\text{position}}(\cdot) = \begin{cases} x_i(n_i) + v_i(n_i)\Big[t_i(n_i+1) - t_i(n_i)\Big] + 0.5a_i(n_i)\Big[t_i(n_i+1) - t_i(n_i)\Big]^2, & \text{for case 1} \\ x_i(n_i) + d_{\text{var}} + d_{\text{ct}}, & \text{for case 2} \end{cases}$$

3.3.5 Prediction of occurrence time of new event

After event $n_i + 1$, the states of the whole network are updated. Let us describe how the occurrence time of a new event is predicted for cybercar i. After event $n_i + 1$, there is still a distance of d_i for cybercar i to go before it arrives at the end of the segment it is running in. Assuming $l_i(n_i+1) = h$ and $s_i(n_i+1) = o$, d_i is given by

- in the case of "**same segment, same link**":

$$d_i = p_{o,h}^{\text{end}} - x_i(n_i+1)$$

- in the case of "**different segments, same link**" or "**different links**":

$$d_i = L_{o,h}$$

If cybercar i can reach the equilibrium speed $V\left(\rho_{o,h}(n_{o,h})\right)$ from $v_i(n_i+1)$ before arriving the end of segment q of link h , that is if

$$d_i > \frac{V_{o,h}^2\left(\rho_{o,h}(n_{o,h})\right) - v_i^2(n_i+1)}{2a_i(n_i+1)}$$

then the predicted occurrence time of the next event cybercar i is given by

$$t_{\text{next},i} \leftarrow t_i(n_i+1) + \frac{V_{o,h}\left(\rho_{o,h}(n_{o,h})\right) - v_i(n_i+1)}{a_i(n_i+1)} + \frac{d_i - \tilde{d}_i}{V_{o,h}\left(\rho_{o,h}(n_{o,h})\right)}$$

with

$$\tilde{d}_i = \frac{V_{o,h}^2\left(\rho_{o,h}(n_{o,h})\right) - v_i^2(n_i+1)}{2a_i(n_i+1)}$$

Otherwise, the predicted occurrence time of the next event of cybercar i is given by

$$t_{\text{next},i} \leftarrow t_i(n_i+1) + \tilde{\tau}$$

where $\tilde{\tau}$ is the solution to

$$v_i(n_i+1)\tilde{\tau} + 0.5a_i(n_i+1)\tilde{\tau}^2 = d_i$$

Let

$$\tilde{a} = 0.5a_i(n_i+1), \quad \tilde{b} = v_i(n_i+1), \quad \tilde{c} = -d_i$$

then we have

$$\tilde{\tau} = \frac{-\tilde{b} + \sqrt{\tilde{b}^2 - 4\tilde{a}\tilde{c}}}{2\tilde{a}}$$

Therefore, there are two cases:

- case 1:

$$d_i > \frac{V_{o,h}^2\left(\rho_{o,h}(n_{o,h})\right) - v_i^2(n_i+1)}{2a_i(n_i+1)}$$

- case 2:

$$d_i \leq \frac{V_{o,h}^2\Big(\rho_{o,h}(n_{o,h})\Big) - v_i^2\big(n_i+1\big)}{2a_i(n_i+1)}$$

and $F_{t_{\text{next}}}(\cdot)$ is given by

$$F_{t_{\text{next}}}(\cdot) = \begin{cases} t_i(n_i+1) + \frac{V_{o,h}\Big(\rho_{o,h}(n_{o,h})\Big) - v_i\big(n_i+1\big)}{a_i(n_i+1)} + \frac{d_i - \tilde{d}_i}{V_{o,h}\Big(\rho_{o,h}(n_{o,h})\Big)}, & \text{for case 1} \\ t_{\text{next},i} \leftarrow t_i(n_i+1) + \tilde{\tau}, & \text{for case 2} \end{cases}$$

3.4 Summary

In this chapter we have considered the modeling of the dynamics and the energy consumption of cybercars in a road network that is only open to cybercars. We have presented two models, namely a discrete-time model and a discrete-event model. In the discrete-time model, the dynamics of cybercars and the states of the network are updated at every simulation time step. Therefore, the discrete-time model is straightforward to derive. In contrast, in the discrete-event model, the dynamics of cybercars and the states of the network are only updated when events occur. Hence, predicting the occurrence of events is involved in the discrete-event model. Actually, the discrete-event model is more time-efficient than the discrete-time model in online computation when the number of cybercars is small. However, we focus on a cybernetic transportation network with a large number of cybercars. Therefore, it is more time-efficient to use the discrete-time model for predicting the dynamics and the energy consumption of cybercars.

Chapter 4

Multi-Agent Dynamic Routing of Cybercars in Cybernetic Transportation Networks

Although the fleet control problem for cybercars can be straightforwardly addressed in a centralized control setting, for reasons of scalability and fast computation, a centralized control method will not be tractable for the control of a large number of cybercars in the large-scale cybernetic transportation systems in the future. In this chapter, we address the dynamic routing of cybercars based on the discrete-time model of the dynamics and the energy consumption of cybercars presented in Chapter 3. We consider minimization of the combined system cost including the total time spent and the total energy consumption by all cybercars and we propose several tractable and scalable multi-agent control methods including multi-agent model predictive control and parameterized control for solving the problem. Finally, experiments by means of numerical simulations illustrate the performance of the proposed control methods.

Parts of this chapter have been included in [81] and [82].

4.1 Introduction

So far, there have been many automated driving technologies available for individual vehicles [16, 129], such as automated lane change [54] and adaptive cruise control [31]. However, there is still no efficient method for the control of a fleet of vehicles. Hence, the large-scale application of cybercars is still hindered. The performance of a cybernetic transportation system depends on the control of cybercars for cooperative routing, collision avoidance, platoon merge and split, etc. Therefore, in order to achieve high performance, the fleet control problem for cybercars, where the cooperation of cybercars is explored by considering all the cybercars in the whole fleet, must be addressed.

Actually, the fleet control problem for cybercars has already been considered in the literature. More specifically, the problem was studied in [7] from a conceptual point of view and a centralized fleet management system for cybercars was proposed. However, that paper only focused on the design of the system architecture without addressing the fleet control problem explicitly. In [13], a novel open-control concept that merges both centralized and decentralized control approaches for cybercars was proposed. However,

that paper only focused on demonstrating how a cybernetic transportation system may benefit from the open-control concept in dealing with perturbations caused by the environment, while it did not introduce a specific algorithm for fleet control of cybercars. In [50], a specific instance of the fleet control problem for cybercars, i.e., the vehicle routing problem for an on-demand transportation system, was studied by representing the road network as a weighted complete graph and defining fixed attributes (travel time, travel cost, etc) for each arc. However, that paper focused on solving the vehicle routing problem on a daily basis without considering the real-time conditions of the network. In contrast, in this chapter, we explore the dynamic routing problem of cybercars by considering the dynamics and the energy consumption of cybercars according to the real-time conditions of the road network. We develop efficient strategies for the dynamic routing of cybercars so that the total costs for all cybercars, including the total time spent (TTS) and the total energy consumption (TEC), are minimized.

By directly incorporating system constraints as inequalities in the control problem formulation, model predictive control (MPC) has shown to be promising for control of road traffic networks [48, 57, 69]. However, for reasons of scalability and fast computation, centralized model predictive control will not be tractable for the control of large-scale cybernetic transportation systems. Therefore, multi-agent control methods have to be employed.

In multi-agent model predictive control, the overall control problem is first divided into a set of subproblems, which are assigned to different agents. The agents then determine their control actions by solving their local subproblems and coordinating with others [23]. Multi-agent MPC algorithms have been applied to power generation systems [128], chemical processes [119], temperature regulation systems [94], and supply chains [86]. In this chapter, the parallel multi-agent model predictive control scheme presented in [95] is adapted and applied to the dynamic routing of cybercars.

Besides, in parameterized control, the control laws are parameterized and then optimized over the parameters with respect to the overall performance of the whole system for a set of representative scenarios. Parameterized control methods have been applied to control of freeway traffic [135], of robotic systems [97], and of baggage handling systems [123]. In this chapter, several computationally fast and scalable multi-agent parameterized control methods are proposed for the dynamic routing of cybercars.

With respect to the literature, the main contributions of this chapter are addressing a specific instance of the fleet control problem of cybercars, i.e., the dynamic routing of a fleet of cybercars, and proposing several tractable and scalable multi-agent control methods to solve the problem.

This chapter is organized as follows. In Section 4.2, the dynamic routing problem of cybercars is formulated. In Section 4.3 and Section 4.4, we propose a multi-agent model predictive control method and six different parameterized control methods for the dynamic routing of cybercars, respectively. Section 4.5 presents a simulation study where the performance of all proposed methods is assessed and compared. Finally, in Section 4.6, the main contributions of this chapter are summarized, and some ideas of future work are presented.

4.2 Model predictive dynamic routing

Based on the discrete-time model presented in Chapter 3, we adopt a model predictive control scheme to formulate the dynamic routing problem of cybercars.

During $[kT,(k+N_\mathrm{p})T]$, the total time spent and the total energy consumption by all cybercars are given by

$$J_\mathrm{TTS}(k)=\sum_{i\in I(k,N_\mathrm{p})}\min\Big((k+N_\mathrm{p})T-T_{\mathrm{start},i},\ T_{\mathrm{stop},i}-kT,\ T_{\mathrm{stop},i}-T_{\mathrm{start},i},\ N_\mathrm{p}T\Big)+J^\mathrm{end}_\mathrm{TTS}(k) \quad (4.1)$$

$$J_\mathrm{TEC}(k)=\sum_{h=0}^{N_\mathrm{p}-1}\sum_{i\in I(k,N_\mathrm{p})}E_i(k+h)\ +J^\mathrm{end}_\mathrm{TEC}(k) \quad (4.2)$$

where N_p denotes the prediction horizon and $I(k,N_\mathrm{p})$ denotes the set of cybercars in the network during $[kT,(k+N_\mathrm{p})T]$, $J^\mathrm{end}_\mathrm{TTS}(k)$ and $J^\mathrm{end}_\mathrm{TEC}(k)$ denote respectively estimates of the expected remaining total time spent and the expected remaining total energy consumption by the cybercars still in the network at $t=(k+N_\mathrm{p})T$ from their positions at $t=(k+N_\mathrm{p})T$ to their destinations. One possible way to obtain $J^\mathrm{end}_\mathrm{TTS}(k)$ and $J^\mathrm{end}_\mathrm{TEC}(k)$ is by using the speeds of the cybercars still in the network at $t=(k+N_\mathrm{p})T$ and considering the shortest time routes for those cybercars computed using e.g. *Dijkstra's algorithm* [40] based on their speeds at $t=(k+N_\mathrm{p})T$.

In order to properly balance $J_\mathrm{TTS}(k)$ and $J_\mathrm{TEC}(k)$, who have possibly different units and different orders of magnitude, the overall objective function is designed as

$$J(k)=w_1\frac{J_\mathrm{TTS}(k)}{J_\mathrm{TTS,typical}}+w_2\frac{J_\mathrm{TEC}(k)}{J_\mathrm{TEC,typical}} \quad (4.3)$$

where $J_\mathrm{TTS,typical}$ and $J_\mathrm{TEC,typical}$ denote typical values[1] of the total time spent and the total energy consumption of all cybercars in one prediction period while w_1,w_2 are nonnegative weights. Note that the control variable $r_i(k)$ for each cybercar i with $i\in I(k,N_\mathrm{p})$ is the route to be selected from a finite set $R_i(k)$ of possible routes[2] from its current position to its destination. Once the route $r_i(k)\in R_i(k)$ is determined, the link sequence following $r_i(k)$ can be determined and then used as input for the model presented in Chapter 3.

Since the dynamics of cybercars are nonlinear and the control variables are discrete, this results in a *Nonlinear Integer Programming* problem. Although there are several algorithms, such as genetic algorithm [100], simulated annealing [100] and DIRECT [63], available for solving this problem, in general, this problem is computationally very hard to solve, especially when the number of cybercars is large. For reasons of scalability and fast computation, a major challenge of achieving dynamic routing for a large fleet of cybercars is to find efficient approximate solution methods. In the next two sections, we propose efficient solution methods for the dynamic routing of cybercars.

[1] These values could e.g., be the averaged total time spent and the averaged total energy consumption of cybercars over all prediction periods in a simulation where the routes of cybercars are predefined or a simple routing strategy (e.g., shortest time route) is used.

[2] This set of routes could be obtained using a K-shortest path algorithm [44] to find a fixed a number of possible routes from the end of the current link of cybercar i to its destination.

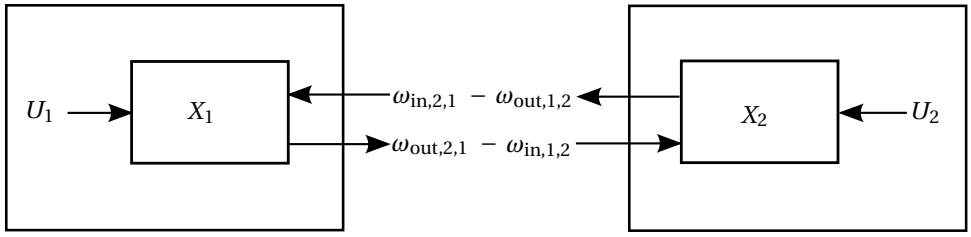

Figure 4.1: Illustration of the interconnection between two subnetworks

4.3 Multi-agent model predictive dynamic routing

In this section, we propose a multi-agent model predictive control method for the dynamic routing of cybercars according to standard multi-agent MPC methods from literature. Given the similarities between the cybernetic transportation network considered in this chapter and the transportation networks considered in [95], we adapt the parallel multi-agent model predictive control scheme presented in [95].

4.3.1 Decomposing the overall network

The whole network is divided into a set G of subnetworks[3]. For each subnetwork, an agent is assigned. At every control step, for each cybercar i, the sequence of subnetworks that will be visited by a cybercar i is first extracted from the shortest-time route from its current position to its destination computed by a shortest path algorithm based on the current traffic condition. After that, the exact route of the cybercar through each subnetwork is determined by the corresponding agent.

For each subnetwork g with a set of neighboring subnetworks $P_g = \{p_{g,1}, ..., p_{g,n_g}\}$, we define:

- $X_g(k)$: local state at time kT including positions, speeds, links and segments of cybercars in the local subnetwork g as well as the traffic densities and the blocking signals of all the links in the local subnetwork
- $U_g(k)$: local control variables i.e., routes of cybercars in the local subnetwork g
- $\omega_{\text{in},g}(k) = [\omega^{\text{T}}_{\text{in},p_{g,1},g}(k), ..., \omega^{\text{T}}_{\text{in},p_{g,n_g},g}(k)]^{\text{T}}$: external inputs from neighboring subnetworks $p_{g,1}, ...p_{g,n_g}$ including the indices, entering points, entering times of the cybercars from the neighboring subnetworks to the local subnetwork
- $\omega_{\text{out},g}(k) = [\omega^{\text{T}}_{\text{out},p_{g,1},g}(k), ..., \omega^{\text{T}}_{\text{out},p_{g,n_g},g}(k)]^{\text{T}}$: outputs to neighboring subnetworks including the indices, exit points, and exit times of the cybercars from the local subnetwork to its neighboring subnetworks

As an example, the interconnection of two subnetworks is illustrated in Figure 4.1.

[3]Note that dividing a network into subnetworks, for which an efficient algorithm has been proposed by [47], is outside the scope of our work. We assume that the network and its division are given, and we only focus on the design of the multi-agent model predictive dynamic routing method.

4.3.2 MPC of a single subnetwork

We assume that at time kT, agent g has full knowledge of the current state of its own subnetwork and of the cybercars in its own subnetwork. Then, given the external inputs $\omega_{\text{in},g}(k)$ from neighboring subnetworks, agent g predicts the future local states using a local model. Given the predicted future local states, we define the following local objective function for agent g at time kT:

$$J_{\text{TTS},g}(k) = \sum_{i\in\Omega_g(k)\cup\Omega_{g,\text{in}}(k)} \min\Big((k+N_\text{p})T - T_{\text{start},g,i},\ N_\text{p}T,\ T_{\text{stop},i} - T_{\text{start},g,i},\ T_{\text{stop},i} - kT, T_{\text{cross},g,i} - T_{\text{start},g,i},\ T_{\text{cross},g,i} - kT\Big) + J^{\text{leave}}_{\text{TTS},g}(k) \tag{4.4}$$

$$J_{\text{TEC},g}(k) = \sum_{h=0}^{N_\text{p}-1} \sum_{i\in\Omega_g(k)\cup\Omega_{g,\text{in}}(k)} E_i(k+h) + J^{\text{leave}}_{\text{TEC},g}(k) \tag{4.5}$$

$$J_g(k) = w_1 \frac{J_{\text{TTS},g}(k)}{J_{\text{TTS,typical}}} + w_2 \frac{J_{\text{TEC},g}(k)}{J_{\text{TEC,typical}}} \tag{4.6}$$

where $\Omega_g(k)$ denotes the set of cybercars in subnetwork g at time kT, $\Omega_{g,\text{in}}(k)$ denotes the set of cybercars entering subnetwork g from neighboring subnetworks during $[kT,(k+N_\text{p})T]$. More specifically, $\Omega_{g,\text{in}}(k)$ is extracted from the interconnected input $\omega_{\text{in},g}(k)$. Furthermore, $T_{\text{cross},g,i}$ denotes the time when car i leaves the subnetwork of agent g and enters another subnetwork, $J^{\text{leave}}_{\text{TTS},g}(k)$ and $J^{\text{leave}}_{\text{TEC},g}(k)$ are estimates of the expected remaining total time spent and the expected remaining total energy consumption by the cybercars still in subnetwork g at time $(k+N_\text{p})T$, from $(k+N_\text{p})T$ to the time they leave the local subnetwork.

Finally, the following local control problem is solved by agent g:

$$\min_{\{r_i(k)\,|\,i\in\Omega_g(k)\cup\Omega_{g,\text{in}}(k)\}} J_g(k) \tag{4.7}$$

subject to

$$r_i(k) \in R_{i,g}(k) \tag{4.8}$$

where $R_{i,g}(k)$ is the finite set of possible routes for cybercar i to go through subnetwork g.

4.3.3 Multi-agent model predictive dynamic routing method

Considering the interconnections among subnetworks and given the formulation of the local control problem of every subnetwork, we adapt the parallel multi-agent model predictive control scheme presented in [95] and apply it to the dynamic routing of cybercars. More specifically, the interconnecting constraints among subnetworks are removed from the constraint set and added to the objective function in the form of additional penalties based on an augmented Lagrangian formulation of the overall control problem. By using such an approach, the formulated problem becomes separable and can then be distributed over the agents. At each control step, the agents solve their local problems iteratively for fixed Lagrange multiplers, followed by updating the Lagrange

multipliers using local solutions. The iterations stop when the Lagrange multipliers do not change anymore or the maximum allowed number of iterations is reached. After that, the agents implement the control actions until the next control step, after which the whole procedure is repeated.

4.4 Parameterized dynamic routing

Determining the routes for all cybercars by solving an online optimization problem requires a huge computational effort. Therefore, in order to provide a balanced trade-off between control performance and computational effort, we propose the use of parameterized control methods. The main idea of parameterized control is to parameterize the control decision-making process and to optimize the parameters of the control laws by solving an optimization problem considering the performance of the control method, see [97] and [123]. After that, the control inputs are determined by using the parameterized control laws with the optimized parameters.

In parameterized dynamic routing of cybercars, the route selection process for cybercars is described using a parameterized control law that is a function of the state of the network. The parameters are optimized so as to optimize the routing performance including the total time spent and the total energy consumption of cybercars. After that, at each control cycle, the route of each cybercar is updated by selecting a route from a limited set of possible routes from its current position to its destination.

At any time step, a finite set of possible routes for each cybcar from its current position to its destination can be generated by using the current state of the network and by using shortest-route algorithms. More specifically, before a shortest-route algorithm is called to generate the limited sets of possible routes for cybercars, the estimated cost on each link j based on the current state of the network is determined by:

$$c_j = \lambda_1 \frac{L_{\text{link},j}}{L_{\text{link,ave}}} + \lambda_2 \frac{1}{T_{\text{link,ave}}} \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\left(\rho_{m,j}(k)\right)} \tag{4.9}$$

where $L_{\text{link},j}$ denotes the length of link j, $L_{\text{link,ave}}$ denotes the average of $L_{\text{link},j}$ over all links, $T_{\text{link,ave}}$ denotes the average link travel time over all links, $M_{\text{segment}}(j)$ denotes the number of segments in link j, λ_1 and λ_2 are given constants. One way to determine $T_{\text{link,ave}}$ is given by:

$$\bar{v}_{\text{free}} = \frac{\sum_{j=1}^{M_{\text{link}}} \sum_{m=1}^{M_{\text{segment}}(j)} v_{\text{free},m,j}}{\sum_{j=1}^{M_{\text{link}}} M_{\text{segment}}(j)}$$

$$T_{\text{link,ave}} = \frac{L_{\text{link,ave}}}{\gamma \cdot \bar{v}_{\text{free}}}$$

where M_{link} denotes the number of links in the network, $\bar{v}_{\text{free}}$ denotes the average free-flow speed over all segments in all links, and γ is a model parameter.

In the following sections, we present the designs of six different parameterized control methods for dynamic routing of cybercars.

4.4.1 Parameterized control method 1

We define $R_i(k)$ as the limited set of possible routes of cybercar i generated at time kT. After that, for each $r \in R_i(k)$, we define $L_{\text{route}}(r)$ as the length of route r, $T_{\text{route}}(r,k)$ as the estimated travel time on route r, and $N_{\text{route}}(r,k)$ as the estimated number of cybercars on route r.

Since the length of each link is fixed, the length of the route r can be easily calculated by summing up of the lengths of all the links belonging to route r. However, even if a route r is given, the travel time and the number of cybercars on that route are still time-dependent. Therefore, at any time when $T_{\text{route}}(r,k)$ and $N_{\text{route}}(r,k)$ are used, they have to be calculated based on the current states of all cybercars and of the network. We propose three approaches to estimate $T_{\text{route}}(r,k)$ and $N_{\text{route}}(r,k)$:

- Approach 1: Only use the current state of the network:

$$T_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\left(\rho_{m,j}(k)\right)} \tag{4.10}$$

$$N_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} N_{m,j}(k) \tag{4.11}$$

  where $\rho_{m,j}(k)$ and $N_{m,j}(k)$ respectively denote the traffic density and number of cybercars in the m-th segment of link j at time step k.

- Approach 2: Predict the future states of the network assuming all cybercars follow the current routes and using the simulation model:

$$\bar{\rho}_{m,j}(k) = \sum_{l=1}^{N_{\text{p}}} \frac{\rho_{m,j}(k+l)}{N_{\text{p}}} \tag{4.12}$$

$$T_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\left(\bar{\rho}_{m,j}(k)\right)} \tag{4.13}$$

$$N_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=0}^{N_{\text{p}}-1} \frac{N_{m,j}(k+l)}{N_{\text{p}}} \tag{4.14}$$

  where $\bar{\rho}_{m,j}(k)$ denotes the average traffic density on the m-th segment of link j over $[kT,(k+N_{\text{p}})T]$.

- Approach 3: Predict the future states of the network assuming all cybercars follow the current routes. In this approach, $N_{\text{route}}(r,k)$ is estimated in the same way as in approach 2. However, different from approach 2, in this approach, $T_{\text{route}}(r,k)$ is estimated by

$$T_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=0}^{N_{\text{p}}-1} \frac{L_{m,j}}{V_{m,j}\left(\rho_{m,j}(k+l)\right)} \frac{1}{N_{\text{p}}} \tag{4.15}$$

Next, at time step k, for each cybercar i in subnetwork $g \in G$, we define a function for

each $r \in R_i(k)$:

$$\varphi_i\big(r,\theta_g,k\big) = \theta_{g,1}\cdot\frac{L_{\text{route}}(r,k)}{L_{\text{route,ave},i}(k)} + \theta_{g,2}\cdot\frac{T_{\text{route}}(r,k)}{T_{\text{route,ave},i}(k)} + \\ + \theta_{g,3}\cdot\frac{N_{\text{route}}(r,k)}{N_{\text{route,ave},i}(k)+\kappa} \tag{4.16}$$

where $\theta_{g,1}$, $\theta_{g,2}$, and $\theta_{g,3}$ are the parameters for subnetwork g and $L_{\text{route,ave},i}(k)$, $T_{\text{route,ave},i}(k)$, and $N_{\text{route,ave},i}(k)$ are respectively the average of $L_{\text{route}}(r,k)$, $T_{\text{route}}(r,k)$, and $N_{\text{route}}(r,k)$ over all $r \in R_i(k)$ for cybercar i, and κ is a small positive number added to the denominator to prevent division by 0. The route of each cybercar i in subnetwork g at kT is then selected as

$$r_i^* = \arg\min_{r\in R_i(k)} \varphi_i\big(r,\theta_g,k\big) \tag{4.17}$$

where $\theta_g = \begin{bmatrix} \theta_{g,1} & \theta_{g,2} & \theta_{g,3} \end{bmatrix}^{\mathrm{T}}$.

4.4.2 Parameterized control method 2

In this method, we first define H_n as the set of outgoing links from node n and $R_{n,d}(k)$ as the limited set of possible routes from node n to node d generated at time kT. After that, for each $j \in H_n$, we define L_j as the length of link j, and $T_{\text{link}}(j,k)$ and $N_{\text{link}}(j,k)$ as the estimated travel time and estimated number of cybercars on link j at time kT, respectively.

We propose three approaches to estimate $T_{\text{link}}(j,k)$ and $N_{\text{link}}(j,k)$. More specifically, at time step k, given the current states of all cybercars and the current conditions of the network, $T_{\text{link}}(j,k)$ and $N_{\text{link}}(j,k)$ are estimated as follows:

- Approach 1:

$$T_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k)\big)} \tag{4.18}$$

$$N_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} N_{m,j}(k) \tag{4.19}$$

- Approach 2:

$$T_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\bar{\rho}_{m,j}(k)\big)} \tag{4.20}$$

$$N_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=0}^{N_{\text{p}}-1} \frac{N_{m,j}(k+l)}{N_{\text{p}}} \tag{4.21}$$

- Approach 3:

$$T_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=0}^{N_{\text{p}}-1} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k+l)\big)} \frac{1}{N_{\text{p}}} \tag{4.22}$$

Next, for each $j \in H_n$, we define $\tilde{r}_{j,d_i}$ as the shortest-time route from the end of link j to the destination node d_i. After that, for each cybercar i in an incoming link of node n with the link in subnetwork g, we define the following function:

$$\begin{aligned}\varphi_{i,n}(j,\theta_g,k) = \theta_{g,1}\cdot\frac{L_{\text{link},j}+L_{\text{route}}(\tilde{r}_{j,d_i},k)}{L_{\text{ave},n,d_i}(k)} \\ +\theta_{g,2}\cdot\frac{T_{\text{link}}(j,k)+T_{\text{route}}(\tilde{r}_{j,d_i},k)}{T_{\text{ave},n,d_i}(k)} \\ +\theta_{g,3}\cdot\frac{N_{\text{link}}(j,k)+N_{\text{route}}(\tilde{r}_{j,d_i},k)}{N_{\text{ave},n,d_i(k)}+\kappa}\end{aligned} \tag{4.23}$$

where d_i denotes the destination of cybercar i, $\theta_{g,1}$, $\theta_{g,2}$ and $\theta_{g,3}$ are parameters for subnetwork g, and L_{ave,n,d_i}, T_{ave,n,d_i}, and N_{ave,n,d_i} are respectively the average of $L_{\text{route}}(r,k)$, $T_{\text{route}}(r,k)$, and $N_{\text{route}}(r,k)$ over all $r \in R_{n,d_i}(k)$.

Finally, the route of each cybercar i in an incoming link of node n and in subnetwork g is selected as follows:

- select the outgoing link from node n as

$$j^* = \arg\min_{j\in H_n}\varphi_{i,n}(j,\theta_g,k) \tag{4.24}$$

  where $\theta_g = \begin{bmatrix}\theta_{g,1} & \theta_{g,2} & \theta_{g,3}\end{bmatrix}^{\mathrm{T}}$.

- the entire route of cybercar i from node n to its destination d_i is selected as:

$$r_i^* = \{j^*\}\cup\tilde{r}_{j^*,d_i} \tag{4.25}$$

4.4.3 Parameterized control method 3

Extended from method 1, parameterized control method 3 determines the route for each cybercar in a sequential way with updated network information (i.e., updated travel time and updated number of cybercars on a route) taking the updated routes of cybercars in the subnetwork into account.

In this method, we define $M_{n,d}(k)$ as the number of cybercars in the incoming links of node n at step k and heading to destination d, and $S_{n,d}(k)$ as the ordered set of the cybercars ordered according to their predicted arrival times at node n (e.g, based on their current speeds and the distance from their current positions to node n). After that, the $M_{n,d}(k)$ cybercars for every n and every d update their routes in the following sequential way:

i) for each $r \in R_{n,d}(k)$, calculate $L_{\text{route}}(r,k)$, $T_{\text{route}}(r,k)$ and $N_{\text{route}}(r,k)$

ii) Let $z = 1$

   a) for the z-th cybercar in $S_{n,d}(k)$, update its route by

$$r_{i(z)}^* = \arg\min_{r\in R_{n,d}(k)}\varphi_{i(z)}(r,\theta_g,k)$$

where $i(z)$ is the global cybercar index that corresponds to the z-th cybercar in $S_{n,d}(k)$.

b) update $T_{\text{route}}\big(r^*_{i(z)},k\big)$ and $N_{\text{route}}\big(r^*_{i(z)},k\big)$ by

$$T_{\text{route}}\big(r^*_{i(z)},k\big) \leftarrow T_{\text{route}}\big(r^*_{i(z)},k\big)\cdot\left(1+\frac{1}{L_{\text{route}}\big(r^*_{i(z)}\big)}\right)$$
$$N_{\text{route}}\big(r^*_{i(z)},k\big) \leftarrow N_{\text{route}}\big(r^*_{i(z)},k\big)+1$$

c) if $z < M_{n,d}(k)$, update $z \leftarrow z+1$ and go back to a); otherwise, stop the procedure.

Note that $\varphi_i(\cdot)$ in this method is the same as (4.16).

4.4.4 Parameterized control method 4

Also extended from method 1, parameterized control method 4 determines the splitting rates of the group of cybercars over a limited set of possible routes.

In this method, for all $r \in R_{n,d}(k)$, we first define $L^{\max}_{n,d,\text{route}}(k)$ as the length of the longest route, $T^{\max}_{n,d,\text{route}}(k)$ as the longest estimated travel time following a route, and $N^{\max}_{n,d,\text{route}}(k)$ as the largest number of cars on a route. Then we define

$$\Delta L_{\text{route}}(r,k) = L^{\max}_{n,d,\text{route}}(k) - L_{\text{route}}(r,k)$$
$$\Delta T_{\text{route}}(r,k) = T^{\max}_{n,d,\text{route}}(k) - T_{\text{route}}(r,k)$$
$$\Delta N_{\text{route}}(r,k) = N^{\max}_{n,d,\text{route}}(k) - N_{\text{route}}(r,k)$$

After that, we define a function $\phi_{n,d}\big(\cdot\big)$ as

$$\phi_{n,d}\big(r,\theta_g,k\big) = \theta_{g,1}\cdot\frac{\Delta L_{\text{route}}(r,k)}{L_{\text{ave},n,d}(k)} + \theta_{g,2}\cdot\frac{\Delta T_{\text{route}}(r,k)}{T_{\text{ave},n,d}(k)} + \\ + \theta_{g,3}\cdot\frac{\Delta N_{\text{route}}(r,k)}{N_{\text{ave},n,d}(k)} \tag{4.26}$$

where $\theta_{g,1}$, $\theta_{g,2}$ and $\theta_{g,3}$ are parameters for subnetwork g.

Further, the percentage of the $M_{n,d}(k)$ cybercars choosing route $r \in R_{n,d}(k)$ is determined by

$$P_{n,d}\big(r,\theta_g,k\big) = \frac{\phi_{n,d}\big(r,\theta_g,k\big)}{\sum_{y\in R_{n,d}(k)}\phi_{n,d}\big(y,\theta_g,k\big)} \tag{4.27}$$

Finally, the routes of cybercars in $S_{n,d}(k)$ are updated as follows:

i) the first round$\Big(P_{n,d}\big(r_{\text{first}},\theta_g\big)\cdot M_{n,d}(k)\Big)$ cybercars in $S_{n,d}(k)$ select the first route r_{first} in $R_{n,d}(k)$.

ii) after that, the following round$\Big(P_{n,d}\big(r_{\text{second}},\theta_g\big)\cdot M_{n,d}(k)\Big)$ in $S_{n,d}(k)$ select the second route r_{second} in $R_{n,d}(k)$.

iii) ...

ii) the remaining cybercars in $S_{n,d}(k)$ select the last route r_{last} in $R_{n,d}(k)$.

Note that after $R_{n,d}(k)$ is generated by using (4.9) and by using a shortest route algorithm, all the routes in $R_{n,d}(k)$ are ordered in an increasing sequence based on their costs. Here, $\{r_{\text{first}}, ..., r_{\text{last}}\}$ is the explicit representation of $R_{n,d}(k)$.

4.4.5 Parameterized control method 5

Parameterized control method 5 is extended from method 2 as in the same way method 3 is extended from method 1.

In this method, the $M_{n,d}(k)$ cybercars for every n and every d update their routes in the following sequential way:

i) for each $j \in H_n$, calculate $L_{\text{link}}(j,k)$ and $L_{\text{route}}(\tilde{r}_{j,d_i},k)$, $T_{\text{link}}(j,k)$ and $T_{\text{route}}(\tilde{r}_{j,d_i},k)$, $N_{\text{link}}(j,k)$ and $N_{\text{route}}(\tilde{r}_{j,d_i},k)$.

ii) Let $z = 1$

a) for the z-th cybercar in $S_{n,d}(k)$, update its route by first selecting the outgoing link from node n as:

$$j^* = \arg\min_{j \in H_n} \varphi_{i(z),n}\big(j, \theta_g, k\big)$$

and then set the entire route $r^*_{i(z)} = \{j^*\} \cup \tilde{r}_{j^*,d_i}$

b) update $T_{\text{route}}\big(r^*_{i(z)},k\big)$ and $N_{\text{route}}\big(r^*_{i(z)},k\big)$ by

$$T_{\text{route}}\big(r^*_{i(z)},k\big) \leftarrow T_{\text{route}}\big(r^*_{i(z)},k\big) \cdot \left(1 + \frac{1}{L_{\text{route}}(r^*_{i(z)})}\right)$$
$$N_{\text{route}}\big(r^*_{i(z)},k\big) \leftarrow N_{\text{route}}\big(r^*_{i(z)},k\big) + 1$$

c) if $z < M_{n,d}(k)$, update $z \leftarrow z+1$ and go back to a); otherwise, stop the procedure.

Note that $\varphi_{i,n}(\cdot)$ in this method is the same as (4.23).

4.4.6 Parameterized control method 6

Finally, parameterized control method 6 is extended from method 2 as in the same way method 4 is extended from method 1.

Based on the definition of $L^{\max}_{n,d,\text{route}}(k)$, $T^{\max}_{n,d,\text{route}}(k)$ and $N^{\max}_{n,d,\text{route}}(k)$ in method 4, for each $j \in H_n$, we first define

$$\begin{aligned}\psi_{n,d}\big(j,\theta_g,k\big) = \theta_{g,1} \cdot \frac{L^{\max}_{n,d,\text{route}}(k) - \Big(L_{\text{link}}(j,k) + L_{\text{route}}(\tilde{r}_{j,d})\Big)}{L_{\text{ave},n,d}} \\ + \theta_{g,2} \cdot \frac{T^{\max}_{n,d,\text{route}}(k) - \Big(T_{\text{link}}(j,k) + T_{\text{route}}(\tilde{r}_{j,d})\Big)}{T_{\text{ave},n,d}} \\ + \theta_{g,3} \cdot \frac{N^{\max}_{n,d,\text{route}}(k) - \Big(N_{\text{link}}(j,k) + N_{\text{route}}(\tilde{r}_{j,d})\Big)}{N_{\text{ave},n,d}}\end{aligned} \tag{4.28}$$

where $\theta_{g,1}$, $\theta_{g,2}$ and $\theta_{g,3}$ are parameters for subnetwork g. After that, the percentage of the $M_{n,d}(k)$ cybercars choosing route $\{j\} \cup \tilde{r}_{j,d}$ is determined by

$$P_{n,d}\big(j,\theta_g,k\big) = \frac{\psi_{n,d}\big(j,\theta_g,k\big)}{\sum_{y\in H_n}\psi_{n,d}\big(y,\theta_g,k\big)} \tag{4.29}$$

Finally, given $P_{n,d}\big(j,\theta_g,k\big)$ for all $j \in H_n$, all cybercars in $S_{n,d}(k)$ update their routes in the way same as in method 4.

4.4.7 Tuning the parameters for parameterized control methods

To tune the parameters of the proposed parameterized control methods, we proceed as follows. We define a scenario as a case where the transport service requests including the starting times, the origins and the destinations of all cybercars, are given. Then, the performance of a parameterized control method on a specific scenario of the dynamic routing of cybercars is evaluated by

$$J_o(\boldsymbol{\theta}) = w_1 \cdot \frac{J_{\text{TTS},o}}{J_{\text{TTS,typical,scenario}}} + w_2 \cdot \frac{J_{\text{TEC},o}}{J_{\text{TEC,typical,scenario}}} \tag{4.30}$$

where o is the index of the scenario, $\boldsymbol{\theta} = \left[\begin{array}{ccc} \theta_1^{\mathrm{T}} & \theta_2^{\mathrm{T}} & \ldots \end{array}\right]^{\mathrm{T}}$, $J_{\text{TTS},o}$ and $J_{\text{TEC},o}$ respectively denote the total time spent and the total energy consumption by all cybercars, $J_{\text{TTS,typical,scenario}}$ and $J_{\text{TEC,typical,scenario}}$ respectively denote the typical values[4] of the total time spent and the total energy consumption by all cybercars in a representative scenario.

Finally, given a number N^{scenario} of representative scenarios, the parameters $\boldsymbol{\theta}$ of the parameterized control method are tuned by minimizing the sum of $J_o(\boldsymbol{\theta})$ over the representative scenarios. More specifically, the parameters $\boldsymbol{\theta}$ are tuned by solving the following *nonlinear programming* problem:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \sum_{o=1}^{N^{\text{scenario}}} J_o(\boldsymbol{\theta}) \\ \text{s.t.} \quad & \text{model equations} \end{aligned} \tag{4.31}$$

which is nonconvex and can be solved by using multiple runs of nonconvex optimization algorithms, e.g. genetic algorithm, simulated annealing, pattern search, or sequential quadratic programming [15].

4.5 Simulation study

In this section, we perform simulation experiments to compare and assess the performance of the proposed control methods for dynamic routing of cybercars. We consider the network shown in Figure 4.2, where there are 11 nodes and 18 links. Note that in Figure 4.2, the number next to a node represents the index of the node, and a number next to a link

[4] These values are e.g., the values of total time spent and total energy consumption of all cybercars in a numerical simulation where the routes of all cybercars are fixed or a simple route control strategy (e.g., fastest route) is used.

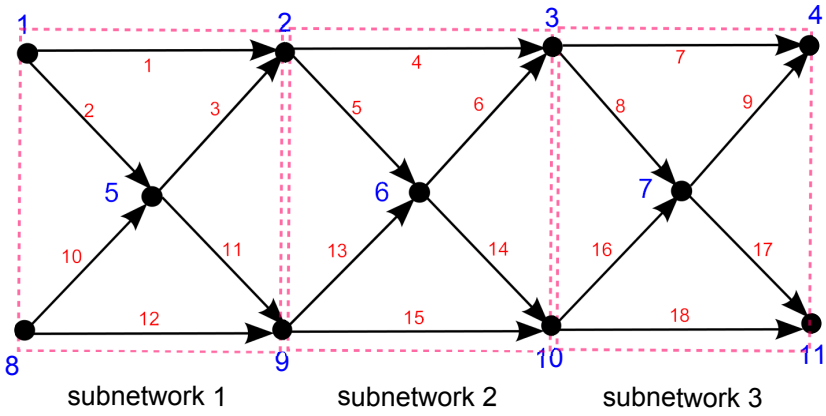

Figure 4.2: Road network used in the case study

Table 4.1: Flow division of cybercars in every scenario

Flows able to update routes			Flows with fixed routes		
Index	O - D	Percentage	Index	O - D	Percentage
1	5 - 7	40%	6	2 - 3	20%
2	1 - 10	15%	7	9 - 10	20%
3	8 - 3	15%	8	2 - 10	30%
4	2 - 11	15%	9	9 - 3	30%
5	9 - 4	15%			

represents the index of the link. Each link is 200 meters long and has 4 segments, with each segment 50 meters long. In the simulations, we generated 30 scenarios, of which 20 are used for tuning the parameters of the proposed control methods and the other 10 are used for evaluating the performance of the proposed control methods. For every scenario o, we set a number $N_{\text{car,enabled}}$ of cybercars that are allowed to update routes and generate a random number $N_{\text{car,fixed}}$ of cybercars with fixed routes. This is done for the reason of considering more cybercars in the network without dramatically increasing the computational complexity of the control problem for centralized MPC and multi-agent MPC. More specifically, for every scenario o, $N_{\text{car,enabled}}$ is determined by

$$N_{\text{car,enabled},o} = 200 + 15 \times \left(\left\lceil \frac{o}{3} \right\rceil - 1 \right)$$

and $N_{\text{car,fixed}}$ is a random integer uniformly distributed in the interval $[100, 200]$. After the numbers $N_{\text{car,enabled}}$ and $N_{\text{car,fixed}}$ are set, the cybercars are divided into 9 flows, which are summarized in Table 4.1. Note that for every three consecutive scenarios starting from $o = 1$, we use the first two for tuning the parameters of the proposed parameterized control methods and use the third one for testing the control performance.

For every scenario, we define the departure times of cybercars for each of the origin-destination flows as follows. For each of the flows 1, 2 and 3, the departure time of the first cybercar is a random number uniformly distributed in the interval $[0.8, 1.6]$ s. In order to create congestion we let the first cybercar of flows 4 and 5 depart later than that of flows 1, 2, and 3, by adding an offset of 40 s. So the departure time of the first cybercar in each of the

flows 4 and 5 is $(40+a)$ s, where a is a random number uniformly distributed in $[0.8, 1.6]$. For the subsequent cybercars in flows 1 to 5 the time interval between the departure times of two consecutive cybercars is a random number uniformly distributed in $[0.8, 1.6]$ s. Besides, for each flow of cybercars with fixed routes, the departure time of the first cybercar is $(1.2+b)$ s, where b a random number uniformly distributed in $[10, 20]$. After that, the time interval between the departure times of two consecutive cybercars is 1.2 s.

The other parameters used in the simulations are: $T = 1$s, $w_1 = 0.7$, $w_2 = 0.3$, $N_\mathrm{p} = 20$, $J_\mathrm{TTS,typical,scenario} = 73202$ s and $J_\mathrm{TEC,typical,scenario} = 11.6767$ kWh, $v_{\mathrm{free},m,j} = 60$ km/h for all m and all j, the length of each cybercar is $L_\mathrm{veh} = 3.2$ m, the mass of each cybercar is $M = 1000$ kg, the efficiency of the electric motor is $\eta_\mathrm{motor} = 0.85$, the round-trip energy recovery coefficient of the electric motor is $\gamma_\mathrm{recover} = 0.38$. Besides, the time interval between two consecutive control steps is $T_\mathrm{c} = 20$ s. The simulations are performed using Matlab 2015a on a cluster computer consisting of 4 blades with 2 eight-core E5-2643 processors, and 3.3 GHz clock rate and 64 GiB memory per blade.

We tuned the parameters for the six proposed parameterized control methods with three different approaches for estimating the travel time and the number of cybercars on a route. For tuning the parameters of each of the 18 combinations, we run the solver *fmincon* of the Matlab Optimization Toolbox with the sequential quadratic programming (SQP) algorithm 60 times using random starting points to solve the nonlinear programming problem (4.31). For simplicity of representation, we refer to the six proposed parameterized dynamic routing methods with their three different estimation approaches as PCx-y, where x is the index of the parameterized dynamic routing method and y denotes the index of the estimation approach, e.g., PC4-3 presents the parameterized dynamic routing method 4 with estimation approach 3. The CPU times for tuning the parameters for the proposed parameterized control methods are in the range of $0.97 \cdot 10^5$ to $1.61 \cdot 10^5$ seconds.

After tuning the parameters, we evaluate the performance of the parameterized control methods on the 10 different testing scenarios. In order to show the effectiveness of the proposed parameterized control methods, we compare the performance of the proposed parameterized control methods on the testing scenarios with those of centralized model predictive control, multi-agent model predictive control, and three greedy control methods using *Dijkstra's Algorithm*. We refer to centralized model predictive control, multi-agent model predictive control, and greedy control methods in the following way:

- C-MPC: centralized MPC using multiple runs of the genetic algorithm with a limited total computation time (i.e., for all runs together) of 3600 s at each control step

- MA-MPC: multi-agent MPC with each agent solving its local problem using bilevel optimization with the following procedure:

  - fix the binary variables and next solve the problem to obtain the real decisions using *fmincon/SQP*
  - solve the binary optimization problem using the genetic algorithm

  with a limited total computation time of 3600 s at each control step. Note that in each local problem of the MA-MPC method, the binary optimization variables are the routes of cybercars. The real optimization variables are the entering times of the cybercars from other subnetworks to the local subnetwork.

Table 4.2: Average online computation times (s) of the control methods

Control methods	Average online computation times (s)
Centralized MPC	69586
Multi-agent MPC	66502
Greedy control	0.07 - 0.14
Parameterized control	0.37 - 4.02

- GC1: greedy control method 1, i.e., shortest distance routing method, using *Dijkstra's algorithm* based on (4.9) with $\lambda_1 = 1$ and $\lambda_2 = 0$
- GC2: greedy control method 2, i.e., shortest time routing method, using *Dijkstra's algorithm* based on (4.9) with $\lambda_1 = 0$ and $\lambda_2 = 1$
- GC3: greedy control method 3, i.e., combined distance and time routing method, using *Dijkstra's algorithm* based on (4.9) with $\lambda_1 = 0.3$ and $\lambda_2 = 0.7$

The average online computation times of all control methods over the testing scenarios are summarized in Table 4.2. For the sake of compactness, only the ranges of average online computation times of the greedy control (GC) methods and the parameterized control (PC) methods are provided.

Since GC3 has the best performance among all the greedy control methods, we use GC3 as the benchmark and calculate the performance improvement of the other control methods for the testing scenarios. More specifically, the performance improvement of a routing method on a specific scenario o compared with that of GC3 on the same scenario is given by

$$P_{\text{im},o} = \frac{J_{\text{GC3},o} - J_o}{J_{\text{GC3},o}} \times 100\%$$

The average performance improvement and the standard deviation of the performance improvement of the routing control methods compared with GC3 over all the testing scenarios are summarized in Table 4.3. In this table, the average performance improvement of a routing control method compared with GC3 over all the testing scenarios is given by

$$\bar{P}_{\text{im}} = \frac{\sum_{o=1}^{N^{\text{testing}}} P_{\text{im},o}}{N^{\text{testing}}}$$

with N^{testing} denoting the number of testing scenarios, and the standard deviation of the performance improvement of a routing method compared with GC3 over all the testing scenario is given by

$$\sigma_{\text{im}} = \sqrt{\frac{\sum_{o=1}^{N^{\text{testing}}} (P_{\text{im},o} - \bar{P}_{\text{im}})^2}{N^{\text{testing}}}}$$

Note that in the second column, a positive number in a cell indicates that the corresponding control method performs better than GC3 on the average over all the testing

Table 4.3: Average performance improvement and standard deviation of the performance improvement of the other control methods with respect to GC3 for all testing scenarios, where a positive number indicates a better performance

Control method	Performance improvement $\bar{P}_{\text{im}}$	Standard deviation σ_{im}
C-MPC	6.36%	4.4%
MA-MPC	1.45%	1.4%
GC1	-4.20%	3.7%
GC2	-0.71%	2.3%
PC1-1	2.67%	7.7%
PC1-2	4.46%	8.7%
PC1-3	3.45%	7.9%
PC2-1	1.99%	7.5%
PC2-2	2.92%	7.7%
PC2-3	3.29%	8.6%
PC3-1	3.89%	7.7%
PC3-2	2.86%	8.8%
PC3-3	4.26%	8.4%
PC4-1	5.04%	8.1%
PC4-2	3.46%	8.4%
PC4-3	2.95%	8.3%
PC5-1	1.45%	8.3%
PC5-2	3.24%	7.4%
PC5-3	3.18%	7.2%
PC6-1	3.90%	8.3%
PC6-2	1.87%	8.1%
PC6-3	1.36%	8.9%

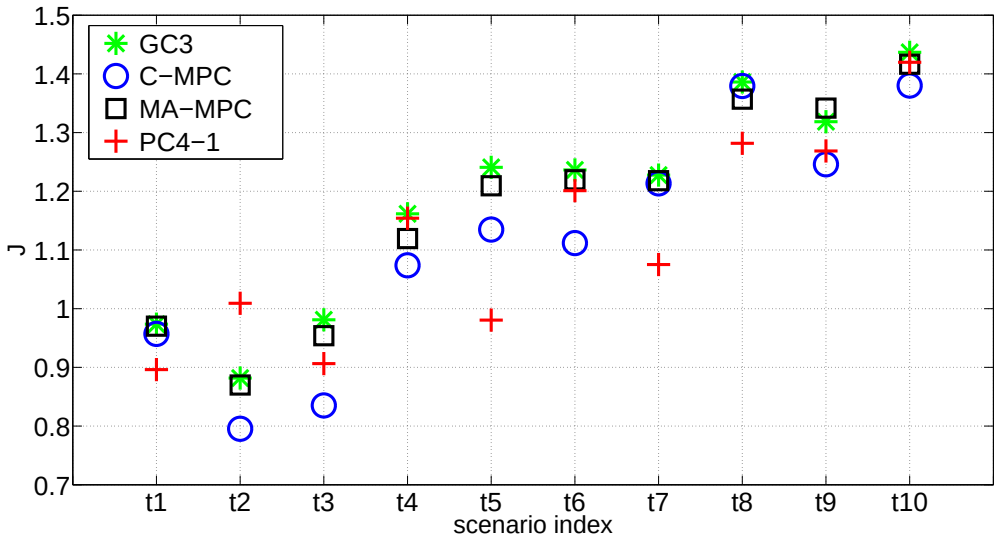

Figure 4.3: Performance of GC3, centralized MPC, multi-agent MPC and PC4-1 for all testing scenarios

scenarios. We found that parameterized control method 4 with estimation approach 1 (i.e. PC4-1, the flow-splitting-rate-based parameterized control method) has the best performance among all the proposed parameterized control methods and it has an average performance improvement of 5.04% on the testing scenarios compared with GC3. Therefore, PC4-1 is selected as the representative of the proposed parameterized control methods to compare further with centralized model predictive control, multi-agent model predictive control, and GC3 on all the testing scenarios. More specifically, Figure 4.3 shows the performance of centralized model predictive control, multi-agent model predictive control, GC3, and PC4-1 for all the testing scenarios.

Actually, we are performing state feedback routing control of cybercars with the proposed parameterized control methods. Different from the prediction of future state of the network in model predictive control where the decision variables are free, in estimation approach 2 and approach 3, the future state of the network are estimated assuming the routes of all cybercars are fixed. If the updated routes of the cybercars are much different from the fixed ones for estimating the future states of the network, then the actual state of the network after the control step could be very different from the estimated state obtained by estimation approach 2 and approach 3. Therefore, even though only the current state of the network are used in estimation approach 1, there is no guarantee that estimation approach 2 and approach 3 perform better than estimation approach 1 in estimating the travel time and the number of cybercars on the routes in the network. Hence, there is no guarantee that a proposed parameterized control method with estimation approach 2 or approach 3 would have better performance in routing control of cybercars than that with estimation approach 1.

Further, it is seen from Table 4.3 that the centralized model predictive control method perform the best on the testing scenarios with an average performance improvement of 6.36% compared with GC3. However, this is achieved by consuming much more computational power, see Table 4.2. For the multi-agent model predictive control method, since the overall problem is an mixed integer nonlinear programming problem, there is no guarantee of convergence of interconnecting variables among subnetworks. In fact, even if given the same computational budget as that of the centralized model predictive control method, the multi-agent model predictive control method does not obtain a performance that is comparable to that of centralized model predictive control. In contrast, the proposed parameterized control method PC4-1 provides a comparable average performance to that of the centralized model predictive control method on the testing scenarios with much less online computation time, also see Table 4.2. For the standard deviations of the performance improvement compared with GC3, those of the centralized model predictive control method and the multi-agent model predictive control method are smaller than those of the parameterized control methods. That is because the centralized model predictive control method and the multi-agent model predictive control method use online optimization for every scenario while the parameters of the parameterized control methods are tuned based on representative scenarios and then fixed for online use.

Finally, it is seen from Figure 4.3 that PC4-1 performs better than GC3 on 9 of the 10 testing scenarios and performs better than the multi-agent model predictive control method on 7 of the testing scenarios. Besides, the centralized model predictive control method only performs better than PC4-1 on 6 of the testing scenarios while on the other 4 it is outperformed by PC4-1. Moreover, it has to be noted that the average online computation time of PC4-1 for all the testing scenarios is only 0.42 s. Therefore, the parameterized

control method PC4-1 is an efficient method for the dynamic routing of a fleet of cybercars.

4.6 Summary

We have addressed the dynamic routing problem of a fleet of cybercars considering the dynamics and the energy consumption of every cybercar according to the real-time conditions of the road network. To minimize the total cost for all cybercars, i.e. a combination of the total time spent and the total energy consumption, we have developed tractable and scalable multi-agent control methods including multi-agent model predictive control and parameterized control. Numerical simulation results indicate that the flow splitting rate based parameterized control method shows comparable control performance to that of centralized model predictive control while requiring much less online computation time. Besides, the flow-splitting-rate-based parameterized control method can be easily applied to road networks with arbitrary topology. Therefore, the flow-splitting-rate-based parameterized control method is effective in solving the dynamic routing problem of a fleet of cybercars.

In our future work, we will first focus on increasing the computational efficiency of the proposed control methods by investigating different levels of model aggregation. Furthermore, we will consider other fleet control problems of cybercars, in particular the problem of assigning transport service requests to cybercars.

Chapter 5

Efficient Routing of Traffic Flows for Urban Transportation Networks

In order to satisfy the increasing demand for road transportation services and to mitigate the problems caused by increasing road traffic, effective and efficient traffic control strategies for urban transportation networks are highly required. Dynamic traffic routing, which selects routes for the traffic flows according to the real-time conditions of the network, is a promising control strategy. In this chapter we propose two novel multi-agent control approaches for dynamic traffic routing in urban transportation networks, namely, hierarchical traffic routing based on network division and bi-level traffic routing based on merging nodes and links. We also illustrate the proposed control approaches for dynamic traffic routing using numerical simulation case studies.

Parts of this chapter have been included in [80].

5.1 Introduction

A number of research works exist on the topic of dynamic traffic routing [10, 26, 30, 68, 87, 88, 102, 103, 126]. More specifically, in [107] the dynamic traffic routing problem was formulated assuming that flows of vehicles select minimum-time routes at each time step and then it was solved using an optimal control approach to minimize the total travel time of all vehicles. However, that paper did not take into account the total energy consumption of the vehicles. In [87, 88], the dynamics of traffic flows in a network is modeled explicitly using a detailed traffic simulator i.e., DYNASMART developed in [89], and a simulation-based approach is proposed for solving the dynamic traffic routing problem. More specifically, the simulation-based approach focuses on traffic simulations and may require much computational efforts. In [26], the properties of mathematical programming models of time-varying flows were explored, and a linear programming formulation was proposed for the dynamic traffic routing problem based on the first-in-first-out (FIFO) properties of traffic flows on individual links. More specifically, the linear programming formulation has the advantage of being suitable for developing efficient solution methods for dynamic traffic routing from a centralized point of view. However, it may require extensive communication efforts in sensing the state of the network. Besides, in [68] the dynamic traffic routing problem was presented in the format of a discrete-time nonlinear optimal control problem for finding the system optimum. The resulting nonlinear optimal control problem was solved using a feasible-direction

algorithm. Moreover, in [126] the dynamic travel times due to potential traffic congestion were modeled based on queue theory and a dynamic traffic routing approach was proposed to minimize the total travel time of all traffic flows. Due to nonlinearity of the model, the trade-off of solution quality and calculation time of the approach was also discussed. In [10], a METANET-like model was proposed to describe the flows of platoons in Automated Highway Systems, and the optimal dynamic routing problem was approximated by a mixed-integer linear problem. In [102, 103] the response behaviors of the drivers towards real-time routing information were accounted for in the dynamic traffic routing problem. After that a fuzzy control based methodology was solved proposed to solve the problem. In [30] an Ant Colony Routing algorithm was proposed to solve the dynamic traffic routing problem such that a well-balanced trade-off between control performance and computational speed was achieved.

Based on some of the work discussed above, a control center is required for each urban transportation network guiding traffic flows in the network. However, an urban transportation network may consist of a large number of roads and intersections, which requires extensive communication efforts for transferring the states of the network to the traffic control center and results in extensive computation efforts in solving the resulting large-scale routing optimization problem. To solve the problem, we propose two novel multi-agent control approaches for dynamic traffic routing in urban transportation networks, which are hierarchical traffic routing based on network division, and bi-level traffic routing based on merging nodes and links. More specifically, in the hierarchical traffic routing approach, a large-scale network is divided into a group of subnetworks and to each subnetwork a local subnetwork traffic routing controller is assigned. At the high level, the interconnecting flows among subnetworks are determined by a centralized network traffic routing controller. After that, the flows within each subnetwork are determined by the corresponding subnetwork traffic routing controller considering minimization of the total travel cost in the subnetwork and minimization of the difference between the sum of traffic flows on the boundaries of the subnetwork and the flows prescribed by the high-level controller. In the bi-level traffic routing approach, nodes and links of a network are merged into a set of aggregated nodes and a set of aggregated links. To each aggregated node a local traffic routing controller is assigned. At the aggregated level the flows on the aggregated links are determined by a centralized traffic routing controller. At the original level, where the actual network is considered, the flows on the actual links are determined by the local traffic routing controllers by negotiating with neighboring local controllers.

With respect to the literature, the main contribution of our work is proposing two scalable and efficient multi-agent control approaches for dynamic traffic routing. The chapter is organized as follows. Section 5.2 describes the general dynamic traffic routing problem. In Section 5.3, we develop the hierarchical dynamic traffic routing approach. Next, the bi-level dynamic traffic routing approach is presented in Section 5.4. Besides, Section 5.5 illustrates the two proposed approaches using numerical simulation study cases. Finally, in Section 5.6, we summarize the main contributions of the chapter and propose some ideas for future work.

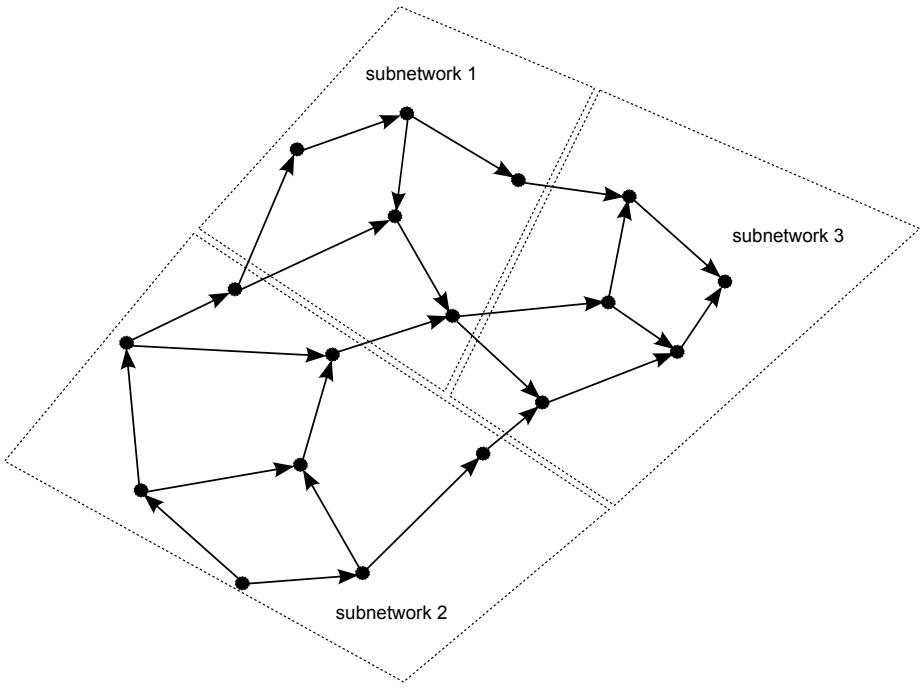

Figure 5.1: An urban transportation network divided into a group of subnetworks

5.2 Problem description

An urban transportation network consists of a set of roads and a set of intersections. For the sake of simplicity, we refer to an intersection as a 'node', and a road section between two intersections as a 'link'. Therefore, an urban transportation network can conceptually be represented by a graph with a set of nodes and a set of links. We consider a discrete-time model of the dynamics of traffic flows in urban transportation networks, where T denotes the length of the simulation time interval and k denotes the simulation step counter. Besides, we assume that all traffic demands originate from source nodes (origins) and target at destination nodes (destinations) and all O-D (origin-destination) traffic demands are obtained via O-D estimation [136]. Finally, we aim to find optimal routes for all O-D traffic demands so that the total travel cost including e.g. the total time spent and the total energy consumption of all the traffic flows is minimized.

5.3 Traffic routing based on network division

In this section, we consider an urban transportation network that has been divided into a group of subnetworks. More specifically, the network is divided by creating the boundaries of the subnetworks on links as shown in Figure 5.1. We consider a high-level network where each subnetwork is mapped to a high-level node and the connection from one subnetwork to another is mapped to a high level-link. Now, we propose a hierarchical traffic routing framework for urban transportation networks:

- The *network traffic routing controller* on the high-level determines traffic flows between subnetworks based on the high-level representation of the network. These flows are computed so that the performance (including total time spent and total

energy consumption of all flows) of the network is optimized. Then these flows are communicated to the *subnetwork traffic routing controllers*.

- Considering that the sum of traffic flows on the boundary of two subnetworks should be as close as possible to the high-level flow between the two subnetworks determined by the *network traffic routing controller*, each *subnetwork traffic routing controller* on the low-level determines the traffic flows on each link in the local subnetwork. The flows within each subnetwork are computed so that the performance (including total time spent and total energy consumption of all flows within the subnetwork and the flow consistency between high level flows and the local flows) of the subnetwork is optimized.

Note that dividing the network is outside the scope of our work. Therefore, we assume that an urban transportation network and its division are given [47], and we only focus on the design of the *network traffic routing controller* and the *subnetwork traffic routing controller*.

5.3.1 Design of the network traffic routing controller

In the fully dynamic case, the travel cost, e.g. travel time and energy consumption, from one subnetwork to another may vary along the time. Accounting for the real-time change of the travel cost directly in the model used for controller design will require high computational efforts in the control decision making procedure. In order to achieve a well-balanced trade-off between control performance and computation speed, we propose to use a quasi-static procedure [30, 120] for the traffic routing controller. More specifically, at each iteration in the quasi-static procedure, a forward simulation is first performed and the travel costs on each link is determined based on the predicted states of the link in the simulation. After that, the travel costs are kept fixed for the control period to let the controller make decisions based on those fixed travel costs. At the end of each iteration, the traffic states are updated and a new iteration starts with updated travel costs. In this section, we present the network traffic routing based on the quasi-static procedure.

Definitions

By letting $\tilde{n}$ denote a high-level node and $\tilde{l}$ denote a high-level link, we define

- $\tilde{t}_{\tilde{l}}^{(z)}$: average travel time on link $\tilde{l}$ at the iteration z in the quasi-static procedure
- $\tilde{e}_{\tilde{l}}^{(z)}$: average energy consumption of a vehicle traveling through link $\tilde{l}$ at the iteration z in the quasi-static procedure
- $\tilde{\mathcal{N}}$: set of high-level nodes
- $\tilde{\mathcal{L}}$: set of high-level links
- $\tilde{\mathcal{D}}$: set of high-level final destinations
- $\tilde{\mathcal{I}}_{\tilde{n}}$: set of incoming links of node $\tilde{n}$
- $\tilde{\mathcal{O}}_{\tilde{n}}$: set of outgoing links of node $\tilde{n}$
- $\tilde{D}_{\tilde{n},\tilde{d}}(k)$: traffic demand from node $\tilde{n}$ to final destination $\tilde{d}$ for interval $\left[kT,(k+1)T\right)$

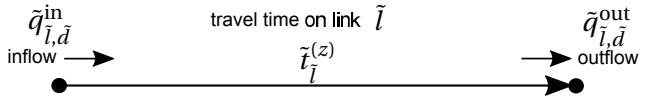

Figure 5.2: Travel time on a link causes a time delay between the inflow and the outflow

- $\tilde{q}^{\text{in}}_{\tilde{l},\tilde{d}}(k)$: flow entering link $\tilde{l}$ traveling towards final destination $\tilde{d}$ for interval $\big[kT,(k+1)T\big)$
- $\tilde{q}^{\text{out}}_{\tilde{l},\tilde{d}}(k)$: flow leaving link $\tilde{l}$ traveling towards final destination $\tilde{d}$ for interval $\big[kT,(k+1)T\big)$
- $\tilde{Q}_{\tilde{n},\tilde{d}}(k)$: queue of traffic towards final destination $\tilde{d}$ but waiting at node $\tilde{n}$ for interval $\big[kT,(k+1)T\big)$
- $q^{\text{cap}}_{\tilde{l}}$: capacity of link $\tilde{l}$

Modeling of dynamics of traffic flows between subnetworks

We model the dynamics of traffic flows between subnetworks by considering flow consistency at each node, time delay between inflow and outflow on each link, non-negativity of traffic flows and traffic queues, and capacity of each link. For modeling the dynamics of traffic flows among subnetworks, we assume that the average travel time $\tilde{t}^{(z)}_{\tilde{l}}$ is an integer multiple of T.

First, at any time step, the sum of outgoing traffic flows from node $\tilde{n}$ is equal to the sum of traffic flows entering node $\tilde{n}$. More specifically, the flow consistency at node $\tilde{n}$ is given by

$$\sum_{j\in\tilde{\mathscr{O}}_{\tilde{n}}} \tilde{q}^{\text{in}}_{j,\tilde{d}}(k)\cdot T + \tilde{Q}_{\tilde{n},\tilde{d}}(k) - \tilde{Q}_{\tilde{n},\tilde{d}}(k-1) = \tilde{D}_{\tilde{n},\tilde{d}}(k)\cdot T + \sum_{j\in\tilde{\mathscr{I}}_{\tilde{n}}} \tilde{q}^{\text{out}}_{j,\tilde{d}}(k)\cdot T,\ \ \forall \tilde{d}\in\tilde{\mathscr{D}},\ \tilde{n}\neq\tilde{d} \tag{5.1}$$

Note that $\sum_{j\in\tilde{\mathscr{O}}_{\tilde{n}}} \tilde{q}^{\text{in}}_{j,\tilde{d}}(k)\cdot T$ is equal to the total traffic flow leaving node $\tilde{n}$ through its outgoing links at time step k, and $\tilde{Q}_{\tilde{n},\tilde{d}}(k) - \tilde{Q}_{\tilde{n},\tilde{d}}(k-1)$ is the difference in queue length at node n. Besides, $\sum_{j\in\tilde{\mathscr{I}}_{\tilde{n}}} \tilde{q}^{\text{out}}_{j,\tilde{d}}(k)\cdot T$ is equal to the total traffic flow entering node n through its incoming links at time step k, and $\tilde{D}_{\tilde{n},\tilde{d}}(k)\cdot T$ is equal to the amount of traffic flow entering node n at time step k due to the traffic demand, .

Second, when a traffic flow enters a link, it will take some time to reach the end of the link, as shown in Figure 6.4. More specifically, the relationship between the incoming flow and the outgoing flow of link $\tilde{l}$ is given by

$$\tilde{q}^{\text{out}}_{\tilde{l},\tilde{d}}(k) = \tilde{q}^{\text{in}}_{\tilde{l},\tilde{d}}\left(k - \frac{\tilde{t}^{(z)}_{\tilde{l}}}{T}\right) \tag{5.2}$$

Since the traffic flow must not be negative and the sum of traffic flows on a link going to different destinations must not exceed the capacity of the link, the flow constraints are given by

$$\tilde{q}^{\text{in}}_{\tilde{l},\tilde{d}}(k) \geq 0 \tag{5.3}$$

$$\sum_{\tilde{d}\in\tilde{\mathscr{D}}} \tilde{q}^{\text{in}}_{\tilde{l},\tilde{d}}(k) \leq \tilde{q}^{\text{cap}}_{\tilde{l}} \tag{5.4}$$

Besides, the queue of traffics waiting at a node must not be negative. Then we have

$$\tilde{Q}_{\tilde{n},\tilde{d}}(k) \geq 0 \tag{5.5}$$

Formulation of traffic routing among subnetworks

Various objectives, e.g., the total time spent and the total energy consumption of all flows and all traffic queues, can be considered. First, the total time spent of all flows and all traffic queues is given by

$$\begin{aligned} J_{\mathrm{TTS}} = &\sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{l}\in\tilde{\mathscr{L}}}\sum_{k=0}^{k_{\mathrm{end}}-1} \tilde{t}_{\tilde{l}}^{(z)} \cdot \tilde{q}_{\tilde{l},\tilde{d}}^{\mathrm{in}}(k)\cdot T + \sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{n}\in\tilde{\mathscr{N}}}\sum_{k=0}^{k_{\mathrm{end}}-1} \tilde{Q}_{\tilde{n},\tilde{d}}(k)\cdot T \\ &+ \sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{l}\in\tilde{\mathscr{L}}} \tilde{\tau}_{\tilde{l},\tilde{d}}\cdot \tilde{q}_{\tilde{l},\tilde{d}}^{\mathrm{in}}(k_{\mathrm{end}})\cdot T + \sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{n}\in\tilde{\mathscr{N}}} \tilde{\iota}_{\tilde{n},\tilde{d}}\cdot \tilde{Q}_{\tilde{n},\tilde{d}}(k_{\mathrm{end}}) \end{aligned} \tag{5.6}$$

where k_{end} denotes the time step corresponding to the end of the simulated period $[0, k_{\mathrm{end}}T]$, $\tilde{\tau}_{\tilde{l},\tilde{d}}$ denotes a measure of the remaining average travel time from link $\tilde{l}$ to destination $\tilde{d}$, and $\tilde{\iota}_{\tilde{n},\tilde{d}}$ denotes a measure of the remaining travel time from node $\tilde{n}$ to destination $\tilde{d}$. One possible way to determine $\tilde{\tau}_{\tilde{l},\tilde{d}}$ and $\tilde{\iota}_{\tilde{n},\tilde{d}}$ is using $\tilde{t}_{\tilde{l}}^{(z)}$ for all links and the shortest time routes computed by *Dijkstra's algorithm* [40]. Second, the total energy consumption of all flows and all traffic queues is given by

$$\begin{aligned} J_{\mathrm{TEC}} = &\sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{l}\in\tilde{\mathscr{L}}}\sum_{k=0}^{k_{\mathrm{end}}-1} \tilde{e}_{\tilde{l}}^{(z)} \cdot \tilde{q}_{\tilde{l},\tilde{d}}^{\mathrm{in}}(k)\cdot T + \sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{n}\in\tilde{\mathscr{N}}}\sum_{k=0}^{k_{\mathrm{end}}-1} \tilde{Q}_{\tilde{n},\tilde{d}}(k)\cdot e_{\mathrm{idle},T} \\ &+ \sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{l}\in\tilde{\mathscr{L}}} \tilde{\chi}_{\tilde{l},\tilde{d}}\cdot \tilde{q}_{\tilde{l},\tilde{d}}^{\mathrm{in}}(k_{\mathrm{end}})\cdot T + \sum_{\tilde{d}\in\tilde{\mathscr{D}}}\sum_{\tilde{n}\in\tilde{\mathscr{N}}} \tilde{\varsigma}_{\tilde{n},\tilde{d}}\cdot \tilde{Q}_{\tilde{n},\tilde{d}}(k_{\mathrm{end}}) \end{aligned} \tag{5.7}$$

where $e_{\mathrm{idle},T}$ denotes the average amount of energy consumed by a vehicle idling for the interval of length T, $\tilde{\chi}_{\tilde{l},\tilde{d}}$ denotes a measure of the remaining energy consumption of a vehicle from the beginning of link $\tilde{l}$ to destination $\tilde{d}$, and $\tilde{\varsigma}_{\tilde{n},\tilde{d}}$ denotes a measure of the remaining energy consumption of a vehicle from node $\tilde{n}$ to destination $\tilde{d}$. One possible way to determine $\tilde{\chi}_{\tilde{l},\tilde{d}}$ and $\tilde{\varsigma}_{\tilde{n},\tilde{d}}$ is using $\tilde{e}_{\tilde{l}}^{(z)}$ for all links and the shortest time routes.

Finally, in order to properly balance J_{TTS} and J_{TEC}, who have possibly different units and different orders of magnitude, the problem to determine the flow fractions on the high-level links is formulated by

$$\begin{aligned} \underset{\tilde{\boldsymbol{q}}^{\mathrm{in}},\tilde{\boldsymbol{q}}^{\mathrm{out}},\tilde{\boldsymbol{Q}}}{\text{minimize}} \quad & J := w_1\frac{J_{\mathrm{TTS}}}{J_{\mathrm{TTS,typical}}} + w_2\frac{J_{\mathrm{TEC}}}{J_{\mathrm{TEC,typical}}} \\ \text{subject to} \quad & \text{flow dynamics over the simulated period} \\ & \text{initial conditions} \end{aligned} \tag{5.8}$$

where the bold symbols $\tilde{\boldsymbol{q}}^{\mathrm{in}}$, $\tilde{\boldsymbol{q}}^{\mathrm{out}}$ and $\tilde{\boldsymbol{Q}}$ are respectively the compact expressions containing $\tilde{q}_{\tilde{l},\tilde{d}}^{\mathrm{in}}(k)$, $\tilde{q}_{\tilde{l},\tilde{d}}^{\mathrm{out}}(k)$ and $\tilde{Q}_{\tilde{l},\tilde{d}}(k)$ for $\tilde{l}\in\tilde{\mathscr{L}}$, $\tilde{n}\in\tilde{\mathscr{N}}$, $\tilde{d}\in\tilde{\mathscr{D}}$ and $k\in\{0,1,...,k_{\mathrm{end}}-1\}$. Besides, $J_{\mathrm{TTS,typical}}$ and $J_{\mathrm{TEC,typical}}$ denote typical values[1] of the total time spent and the total

[1] These values could e.g., be the total time spent and the total energy consumption of all traffic flows in a simulation where the routes of all flows are fixed or where a simple routing strategy (e.g., shortest-time route) is used.

energy consumption of traffic flows and traffic queues for the period of length $k_{\text{end}}T$. Note that the optimization problem (5.8) is a linear programming (LP) problem and it can be solved efficiently by using the well-developed LP algorithms [35] like simplex algorithm, ellipsoid algorithm, or interior-point algorithm.

Quasi-static traffic routing procedure

Given $\tilde{t}_{\tilde{l}}^{(z)}$ and $\tilde{e}_{\tilde{l}}^{(z)}$ for each $\tilde{l} \in \tilde{\mathscr{L}}$, we have formulated the optimal traffic routing among subnetworks. However, considering that the travel time and the energy consumption from one subnetwork to another may change with the traffic states, we propose to solve the dynamic traffic routing problem among subnetworks using the following quasi-static procedure:

i) Initialize $\tilde{t}_{\tilde{l}}^{(1)}$ and $\tilde{e}_{\tilde{l}}^{(1)}$ for each $\tilde{l} \in \tilde{\mathscr{L}}$ based on historical data.

ii) At each iteration z, solve the optimization problem (5.8) to obtain the flows among subnetworks

iii) Update $\tilde{t}_{\tilde{l}}^{(z+1)}$ and $\tilde{e}_{\tilde{l}}^{(z+1)}$ for each $\tilde{l} \in \tilde{\mathscr{L}}$ based on the flows obtained in Step ii)

iv) Stop if the solution to the optimization problem (5.8) converges or the allowed maximum number of iteration is reached; otherwise go back to Step ii).

Note that the models to update $\tilde{t}_{\tilde{l}}^{(z)}$ and $\tilde{e}_{\tilde{l}}^{(z)}$ for each $\tilde{l} \in \tilde{\mathscr{L}}$ based on the predicted traffic states are described next.

Given $q_{\tilde{l},\tilde{d}}^{\text{in}}(1),...,q_{\tilde{l},\tilde{d}}^{\text{in}}(k_{\text{end}})$ and $q_{\tilde{l},\tilde{d}}^{\text{out}}(1),...,q_{\tilde{l},\tilde{d}}^{\text{out}}(k_{\text{end}})$ for all $\tilde{l}$ and $\tilde{d}$ determined by solving the optimization problem (5.8) at the current iteration z, the traffic density in link $\tilde{l}$ towards destination $\tilde{d}$ at time step k is determined by

$$\rho_{\tilde{l},\tilde{d}}(k) = \rho_{\tilde{l},\tilde{d}}(k-1) + \frac{\left(q_{\tilde{l},\tilde{d}}^{\text{in}}(k) - q_{\tilde{l},\tilde{d}}^{\text{out}}(k)\right)\cdot T}{L_{\tilde{l}}} \tag{5.9}$$

where $\tilde{L}_{\tilde{l}}$ denotes the length of link $\tilde{l}$. After that, the traffic density in link $\tilde{l}$ at step k is given by

$$\rho_{\tilde{l}}(k) = \sum_{\tilde{d}\in\tilde{\mathscr{D}}} \rho_{\tilde{l},\tilde{d}}(k) \tag{5.10}$$

Further, the equilibrium speed of vehicles traveling in link $\tilde{l}$ at step k is determined by

$$v_{\tilde{l}}(k) = V_{\tilde{l}}\left(\rho_{\tilde{l}}(k)\right) \tag{5.11}$$

where the function $V_{\tilde{l}}(\cdot)$ describes how the equilibrium speed of vehicles depends on the traffic density. One possible way to determine $V_{\tilde{l}}(\cdot)$ has been presented in Section 3.2.2 of Chapter 3. Then, the average speed of vehicles traveling on link $\tilde{l}$ is given by

$$\bar{v}_{\tilde{l}}^{(z+1)} = \frac{1}{k_{\text{end}}} \sum_{k=0}^{k_{\text{end}}-1} v_{\tilde{l}}(k) \tag{5.12}$$

and the average travel time of link $\tilde{l}$ at next the iteration $z+1$ is given by

$$\tilde{t}_{\tilde{l}}^{(z+1)} = \text{round}\left(\frac{\tilde{L}_{\tilde{l}}}{\bar{v}_{\tilde{l}}^{(z+1)} \cdot T}\right) \cdot T \tag{5.13}$$

After that, $\tilde{e}_{\tilde{l}}^{(z+1)}$ is updated by

$$\tilde{e}_{\tilde{l}}^{(z+1)} = F_{\tilde{l}}\left(\bar{v}_{\tilde{l}}^{(z+1)}, \tilde{t}_{\tilde{l}}^{(z+1)}, L_{\tilde{l}}\right) \tag{5.14}$$

where the function $F_{\tilde{l}}(\cdot)$ is obtained via heuristics or via experimental data. One possible way to define $F_{\tilde{l}}(\cdot)$ has been presented in Section 3.2.6 of Chapter 3.

5.3.2 Design of the subnetwork traffic routing controller

In this section, we present the subnetwork traffic routing approach based on the quasi-static procedure. Note that for the rest of Section 5.3.2, the variables without tilde pertain to links and nodes within each subnetwork and they are defined in the same way as the corresponding variables with tilde for high-level links and high-level nodes in Section 5.3.1.

Model of dynamics of traffic flows within a subnetwork

For modeling the dynamics of traffic flows within a subnetwork, we assume that the average travel time $t_l^{(z)}$ is an integer multiple of T. After that, the flow consistency at each node, the time delay between inflow and outflow on each link, the non-negativity of traffic flows and traffic queues, and the capacity of each link within the subnetwork are modeled in the same way as (5.1)-(6.7).

Formulation of traffic routing within a subnetwork

The total time spent of all traffic flows and traffic queues within a subnetwork g is given by

$$\begin{aligned} J_{\text{TTS},g} = &\sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_g}\sum_{k=0}^{k_{\text{end}}-1} t_l^{(z)} \cdot q_{l,d}^{\text{in}}(k)\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_g}\sum_{k=0}^{k_{\text{end}}-1} Q_{n,d}(k)\cdot T \\ &+ \sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_g} \tau_{l,d}\cdot\cdot q_{l,d}^{\text{in}}(k_{\text{end}})\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_g} \iota_{n,d}\cdot Q_{n,d}(k_{\text{end}}) \end{aligned} \tag{5.15}$$

where $\mathscr{L}_g$ and $\mathscr{N}_g$ denote the set of links and the set of nodes within subnetwork g, respectively. The total energy consumption of all traffic flows and all traffic queues within a subnetwork g is given by

$$\begin{aligned} J_{\text{TEC},g} = &\sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_g}\sum_{k=0}^{k_{\text{end}}-1} e_l^{(z)} \cdot q_{l,d}^{\text{in}}(k)\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_g}\sum_{k=0}^{k_{\text{end}}-1} Q_{n,d}(k)\cdot e_{\text{idle},T} \\ &+ \sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_g} \chi_{l,d}\cdot q_{l,d}^{\text{in}}(k)\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_g} \varsigma_{n,d}\cdot Q_{n,d}(k_{\text{end}}) \end{aligned} \tag{5.16}$$

Besides, considering the sum of traffic flows on the links on the boundary of two subnetworks should be as close as possible to the flows between the two subnetworks determined by the *network traffic routing controller*, the mismatch of the flows at the two levels for subnetwork g is given by

$$J_{\mathrm{FM},g} = \sum_{g^* \in \mathcal{V}_g^{\mathrm{in}}} \left\| \left(\sum_{d \in \mathcal{N}_{\tilde{d}}} \sum_{l \in \mathcal{L}_{g,g^*}} q_{l,d}^{\mathrm{in}}(k) \right) - \tilde{q}_{\tilde{l}_{g,g^*},\tilde{d}}^{\mathrm{in}}(k) \right\|_1 + \sum_{g' \in \mathcal{V}_g^{\mathrm{out}}} \left\| \left(\sum_{d \in \mathcal{N}_{\tilde{d}}} \sum_{l \in \mathcal{L}_{g',g}} q_{l,d}^{\mathrm{in}}(k) \right) - \tilde{q}_{\tilde{l}_{g',g},\tilde{d}}^{\mathrm{in}}(k) \right\|_1 \tag{5.17}$$

where $\mathcal{V}_g^{\mathrm{in}}$ is the set of subnetworks from which subnetwork g has incoming flows, $\mathcal{V}_g^{\mathrm{out}}$ is the set of subnetworks to which subnetwork g has outgoing flows, $\mathcal{L}_{g,g^*}$ is the set of links on the boundary between g and g^* directed to subnetwork g, $\tilde{l}_{g,g^*}$ is the high-level link directed from subnetwork g^* to subnetwork g, and $\tilde{l}_{g',g}$ is the high-level link directed from subnetwork g to subnetwork g'.

Finally, in order to properly balance the three objectives, the traffic routing within a subnetwork g is formulated by

$$\begin{aligned} \underset{\boldsymbol{q}_g^{\mathrm{in}}, \boldsymbol{q}_g^{\mathrm{out}}, \boldsymbol{Q}_g}{\text{minimize}} \quad & J_g := w_1 \frac{J_{\mathrm{TTS},g}}{J_{\mathrm{TTS,typical}}} + w_2 \frac{J_{\mathrm{TEC},g}}{J_{\mathrm{TEC,typical}}} + w_3 \frac{J_{\mathrm{FM},g}}{J_{\mathrm{FM,typical}}} \\ \text{subject to} \quad & \text{local flow dynamics over the simulated period} \\ & \text{initial conditions} \end{aligned} \tag{5.18}$$

where $\boldsymbol{q}_g^{\mathrm{in}}$, $\boldsymbol{q}_g^{\mathrm{out}}$ and $\boldsymbol{Q}_g$ are respectively the compact expressions containing the $q_{l,d}^{\mathrm{in}}(k)$, $q_{l,d}^{\mathrm{out}}(k)$ and $Q_{n,d}(k)$ for $l \in \mathcal{L}_g$, $n \in \mathcal{N}_g$ $d \in \mathcal{D}$ and $k \in \{0, 1, ..., k_{\mathrm{end}} - 1\}$. Besides, $J_{\mathrm{FM,typical}}$ denotes the typical value[2] for the mismatch of traffic flows.

Quasi-static traffic routing within a subnetwork

The traffic routing procedure within a subnetwork is the same as that used for the traffic routing among subnetworks. More specifically, the optimization variables $\boldsymbol{q}_g^{\mathrm{in}}$, $\boldsymbol{q}_g^{\mathrm{out}}$ and $\boldsymbol{Q}_g$ for the traffic routing problem within a subnetwork g are computed in the same way as the one used for computing the optimization variables for the traffic routing problem among subnetworks.

5.4 Bi-level traffic routing based on merging nodes and links

In Section 5.3, we have presented hierarchical traffic routing based on network division. In this section, we propose another alternative multi-agent control approach for traffic routing based on merging nodes and links. Conceptually, merging nodes and links in a network is different from dividing a network into subnetworks. More specifically, when merging nodes and links, nodes are merged first and links are merged automatically, and the topological features of the network are still maintained. In contrast, when dividing a network into

[2]This value could e.g., be the average of mismatch of traffic flows at the two levels for all subnetworks in a simulation where a simple strategy (e.g. shortest-time route) is used separately for traffic flow routing at the two levels.

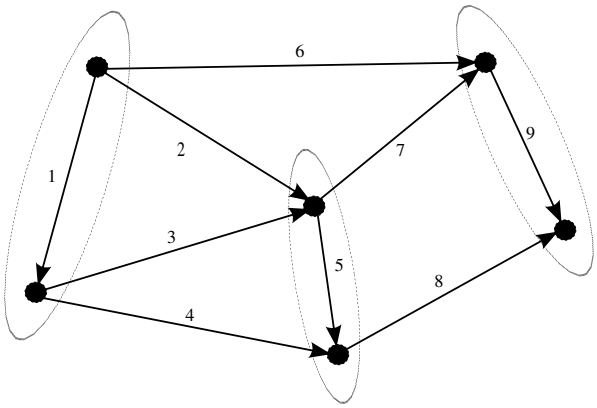

Figure 5.3: Graphical representation of an original road network, where the actual nodes and the actual links within each dotted ellipse are merged into an aggregated node in Figure 5.4. The dotted ellipses are not essential and they are only used to highlight the actual nodes and the actual links that are merged into aggregated nodes.

subnetworks, a large number of nodes and links are grouped simultaneously into a subnetwork, and the topological features of the network may not be maintained.

Different from the hierarchical traffic routing based on network division, for traffic routing based on merging nodes and links, we develop a bi-level control approach where a centralized controller at the aggregated level determines the traffic flows on the aggregated links for the entire aggregated network and a set of local controllers assigned to aggregated nodes cooperatively determine traffic flows on original links considering minimizing the differences between the sums of traffic flows on the original links merged into aggregated links and the flows on the corresponding aggregated links.

5.4.1 Aggregation of nodes and links in a network

Given an urban transportation network conceptually represented by a set of nodes and a set of link, an aggregated network can be generated by grouping nodes based on their geographical positions, their functions, e.g., origins, intermediate nodes, or destinations, and the number of incoming and outgoing links, etc. As an example, the network shown in Figure 5.4 is an aggregated one for the actual network in Figure 5.3. More specifically, due to the merging of nodes, some actual links are merged into aggregated nodes while others are merged into aggregated links. Note that Figure 5.4 shows the aggregated nodes and the aggregated links which of course differ from the actual nodes and the actual links. Also note that generating an aggregated network from an actual network, for which efficient algorithm has been proposed by [25], is outside the scope of our work. Therefore, we assume an actual network and its aggregated network are given, and we only focus on dynamic traffic flow routing on the aggregated level and on the original level. Besides, for the bi-level approach, we refer to the aggregated level as the high level, and refer to the original level as the low level.

5.4.2 Centralized traffic routing at the high level

By defining the variables for aggregated links and aggregated nodes in the same way as the corresponding variables for high-level links and high-level nodes are defined in Section 5.3.1,

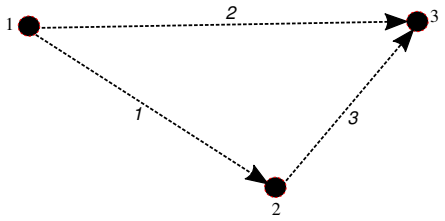

Figure 5.4: Aggregated representation of the road network shown in Figure 5.3

the design of the centralized traffic routing controller at the high level is the same as the design of the *network traffic routing controller* presented in Section 5.3.1.

5.4.3 Distributed traffic routing at the low level

Before formulating the traffic routing problem at the low level, as illustrated in Figure 5.3, we need to consider that some actual links are merged into aggregated nodes while others are merged into aggregated links and that each aggregated link connects to two aggregated nodes. Besides, the flow consistency at the low level within each aggregated node must be maintained and the sum of flows on the actual links merged into each aggregated link should be as close as possible to the flow on the aggregated link. Based on these considerations, we define the flows on the actual links merged in an aggregated link connecting two aggregated nodes as the interconnecting variables between these two aggregated nodes, and propose to assign an agent to each aggregated node focusing on:

- negotiating with other agents on the interconnecting variables,
- maintaining the flow consistency at the low level for each local agent,
- minimizing the total travel cost of all flows for each local agent,
- minimizing the difference between the sum of flows on the original links merged into each aggregated link and the flows on the aggregated links determined on the high level.

Therefore, we first formulate the local traffic routing problem for a local agent. Next, we define the combined overall dynamic flow routing problem at the low level and propose multi-agent control methods to solve the problem. Note that the variables in Section 5.4.3 without tilde pertain to actual links and actual nodes and they are defined in the same way as the corresponding variables with tilde for aggregated links and aggregated nodes.

Interconnection of aggregated nodes

For each aggregated node $\tilde{n}$ with a set of neighboring aggregated nodes $\tilde{\mathcal{N}}_{\tilde{n},\text{neighbor}} = \{\tilde{p}_{\tilde{n},1}, \tilde{p}_{\tilde{n},2}, ..., \tilde{p}_{\tilde{n},M_{\tilde{n}}}\}$, we define:

- $\omega_{\text{in},\tilde{n}} = [\omega^{\text{T}}_{\text{in},\tilde{p}_{\tilde{n},1},\tilde{n}}, \omega^{\text{T}}_{\text{in},\tilde{p}_{\tilde{n},2},\tilde{n}}, ..., \omega^{\text{T}}_{\text{in},\tilde{p}_{\tilde{n},M_{\tilde{n}}},\tilde{n}}]^{\text{T}}$: external inputs from neighboring aggregated nodes including all flows on the actual links merged in all the incoming aggregated links of node $\tilde{n}$

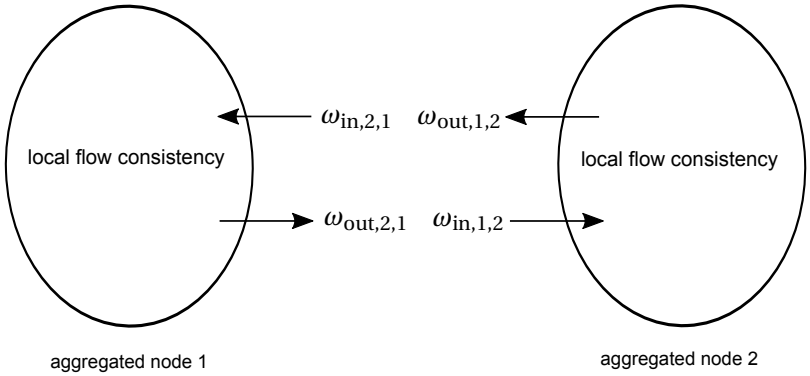

Figure 5.5: Illustration of interconnection between two aggregated nodes

- $\omega_{\text{out},\tilde{n}} = [\omega^{\text{T}}_{\text{out},\tilde{p}_{\tilde{n},1},\tilde{n}}, \omega^{\text{T}}_{\text{out},\tilde{p}_{\tilde{n},2},\tilde{n}}, ..., \omega^{\text{T}}_{\text{out},\tilde{p}_{\tilde{n},M_{\tilde{n}}},\tilde{n}}]^{\text{T}}$: external outputs to neighboring aggregated nodes including all flows on the actual links merged in all the outgoing aggregated links of node $\tilde{n}$

As an example, the interconnection of two aggregated nodes is illustrated in Figure 5.5.

Modeling of dynamics of traffic flows at the low level

At the low level, the flow consistency at each node, the time delay between inflow and outflow on each link, the non-negativity of traffic flows and traffic queues, and the capacity of each link are modeled in the same way as (5.1)-(6.7).

Local problem of an agent

For each aggregated node $\tilde{n}$, the total time spent of local traffic flows and local traffic queues at the low level is given by

$$\begin{aligned} J_{\text{TTS},\tilde{n}} = &\sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_{\tilde{n}}}\sum_{k=0}^{k_{\text{end}}-1} t_l^{(z)}\cdot q^{\text{in}}_{l,d}(k)\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_{\tilde{n}}}\sum_{k=0}^{k_{\text{end}}-1} Q_{n,d}(k)\cdot T \\ &+ \sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_{\tilde{n}}} \tau_{l,d}\cdot q^{\text{in}}_{l,d}(k_{\text{end}})\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_{\tilde{n}}} \iota_{n,d}\cdot Q_{n,d}(k_{\text{end}}) \end{aligned} \tag{5.19}$$

where $\mathscr{L}_{\tilde{n}}$ is the set of links merged into aggregated node $\tilde{n}$ and interconnecting incoming links and interconnecting outgoing links of aggregated node $\tilde{n}$, and $\mathscr{N}_{\tilde{n}}$ is the set of nodes merged into aggregated node $\tilde{n}$. The total energy consumption of local traffic flows and local traffic queues is given by

$$\begin{aligned} J_{\text{TEC},\tilde{n}} = &\sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_{\tilde{n}}}\sum_{k=0}^{k_{\text{end}}-1} e_l^{(z)}\cdot q^{\text{in}}_{l,d}(k)\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_{\tilde{n}}}\sum_{k=0}^{k_{\text{end}}-1} Q_{n,d}(k)\cdot e_{\text{idle},T} \\ &+ \sum_{d\in\mathscr{D}}\sum_{l\in\mathscr{L}_{\tilde{n}}} \chi_{l,d}\cdot q^{\text{in}}_{l,d}(k_{\text{end}})\cdot T + \sum_{d\in\mathscr{D}}\sum_{n\in\mathscr{N}_{\tilde{n}}} \varsigma_{n,d}\cdot Q_{n,d}(k_{\text{end}}) \end{aligned} \tag{5.20}$$

Besides, the agent assigned to aggregated node $\tilde{n}$ should also consider minimizing the difference between the sum of flows on the actual links merged into each aggregated link and the flows on the aggregated links determined on the aggregated level. More specifically,

this objective is given by

$$J_{\mathrm{FM},\tilde{n}} = \sum_{\tilde{d}\in\tilde{\mathscr{D}}} \sum_{\tilde{l}\in\tilde{L}_{\tilde{n}}} \left\| \left(\sum_{d\in\mathscr{N}_{\tilde{d}}} \sum_{l\in\mathscr{L}_{\tilde{l}}} q^{\mathrm{in}}_{l,d}(k) \right) - \tilde{q}^{\mathrm{in}}_{\tilde{l},\tilde{d}}(k) \right\|_1 \tag{5.21}$$

where $\tilde{L}_{\tilde{n}}$ is the set of aggregated incoming links and aggregated outgoing links of aggregated node $\tilde{n}$, and $\mathscr{L}_{\tilde{l}}$ is the set of actual links merged into aggregated link $\tilde{l}$.

Finally, in order to properly balance the three objectives, the local problem for each agent $\tilde{n}$ is formulated by

$$\begin{aligned} \underset{\boldsymbol{q}^{\mathrm{in}}_{\tilde{n}},\boldsymbol{q}^{\mathrm{out}}_{\tilde{n}},\boldsymbol{Q}_{\tilde{n}}}{\text{minimize}} \quad & J_{\tilde{n}} := w_1 \frac{J_{\mathrm{TTS},\tilde{n}}}{J_{\mathrm{TTS,typical}}} + w_2 \frac{J_{\mathrm{TEC},\tilde{n}}}{J_{\mathrm{TEC,typical}}} + w_3 \frac{J_{\mathrm{FM},\tilde{n}}}{J_{\mathrm{FM,typical}}} \\ \text{subject to} \quad & \text{local flow dynamics over the simulated period} \\ & \text{initial conditions} \end{aligned} \tag{5.22}$$

Combined overall control problem on the low level and distributed control methods

The combined overall control problem at the low level is the problem formed by the aggregation of the local problems including the interconnecting constraints for all $\tilde{n} \in \tilde{\mathscr{N}}$:

$$\begin{aligned} \omega_{\mathrm{in},\tilde{p}_{\tilde{n},1},\tilde{n}} &= \omega_{\mathrm{out},\tilde{n},\tilde{p}_{\tilde{n},1}} \\ &\cdots \\ \omega_{\mathrm{in},\tilde{p}_{\tilde{n},M_{\tilde{n}}},\tilde{n}} &= \omega_{\mathrm{out},\tilde{n},\tilde{p}_{\tilde{n},M_{\tilde{n}}}} \end{aligned} \tag{5.23}$$

Generally, standard distributed control methods rom literature [4, 23, 85, 94, 119], e.g. distributed MPC based on dual decomposition and alternative direction method of multipliers, and distributed MPC based on agent negotiation, can be used to solve the combined overall control problem.

Initial values for the interconnecting values for negotiation among agents can be generated from the flows on the aggregated links determined at the high level. By letting the symbol y in $q^{\mathrm{in},(y)}_{l,d}(k)$ denote the negotiation counter in the distributed control approach, the initial value for $q^{\mathrm{in},(y)}_{l,d}(k)$ with $l \in \mathscr{L}_{\tilde{l}}$ can be given by

$$q^{\mathrm{in},(0)}_{l,d}(k) = \frac{\beta_{l,d}}{\sum_{d\in\mathscr{N}_{\tilde{d}}} \sum_{j\in\mathscr{L}_{\tilde{l}}} \beta_{j,d}} \cdot \tilde{q}^{\mathrm{in}}_{\tilde{l},\tilde{d}}(k) \tag{5.24}$$

where $\beta_{l,d}$ is a parameter associated with link l and destination d that represents the rate of traffic flow on link l towards destination d over the total traffic flow on aggregated link $\tilde{l}$ towards aggregated destination $\tilde{d}$. The value of $\beta_{l,d}$ may be determined statistically by using historical data or analytically by taking the features of link l into account, such as the capacity of the link, the average travel time on the link, the length of the link, and the characteristics (e.g., the number of incoming links and the number of outgoing links) of nodes connected to the link, etc.

Quasi-static traffic routing procedure at the low level

We also use a quasi-static decision-making procedure for traffic routing at the low level. This quasi-static procedure is the same as the quasi-static procedure used for the traffic routing

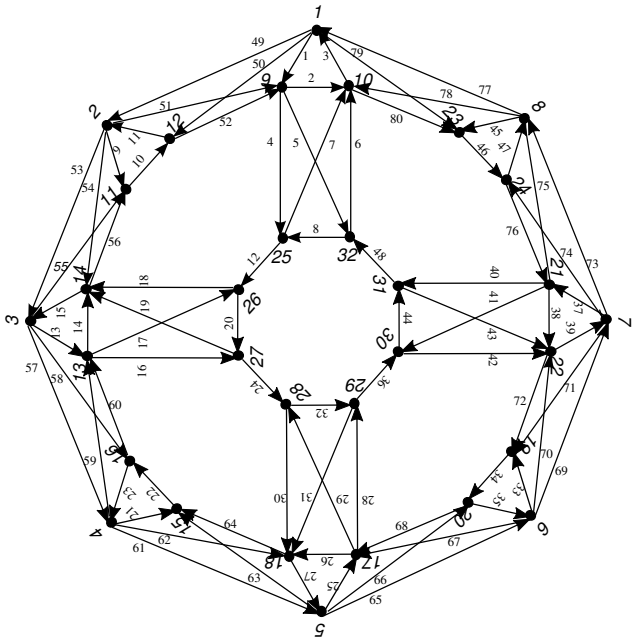

Figure 5.6: The network used in the case study for dynamic traffic routing

Table 5.1: Lengths of links in the representative subnetwork shown in Figure 5.7

link	length (m)	link	length (m)
a, b, c	300	l	300
d, f	600	m, q	800
e, g	700	n, r	700
h	300	o, s	700
i, j, k	300	p, t	500

among subnetworks in Section 5.3.1.

5.5 Simulation study

In this section, we perform simulation experiments to illustrate the two methods proposed above. The simulations are performed using Matlab 2015b on a cluster computer consisting of 4 blades with 2 eight-core E5-2643 processors, and 3.3 GHz clock rate and 64 GiB memory per blade.

5.5.1 Simulation setup

We consider the network shown in Figure 5.6, where there are 32 nodes and 80 links. The network consists of 4 identical substructures. For the sake of simplicity in indicating the lengths of all links in the network, one representative substructure has been shown in Figure 5.7 to help refer to links. The lengths of all links in this subnetwork are summarized in Table 5.1. For hierarchical traffic routing based on network division, we divide the network into 4 subnetworks as shown in Figure 5.8. The resulting high-level network

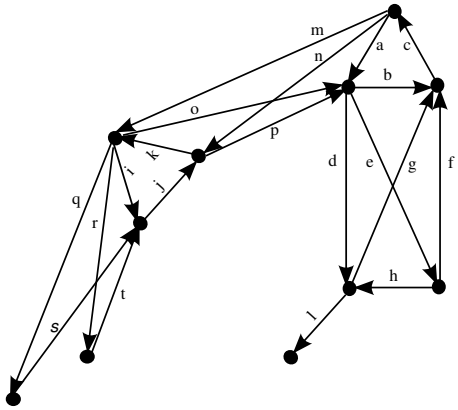

Figure 5.7: Substructure of the subnetwork used to indicate the lengths of all links

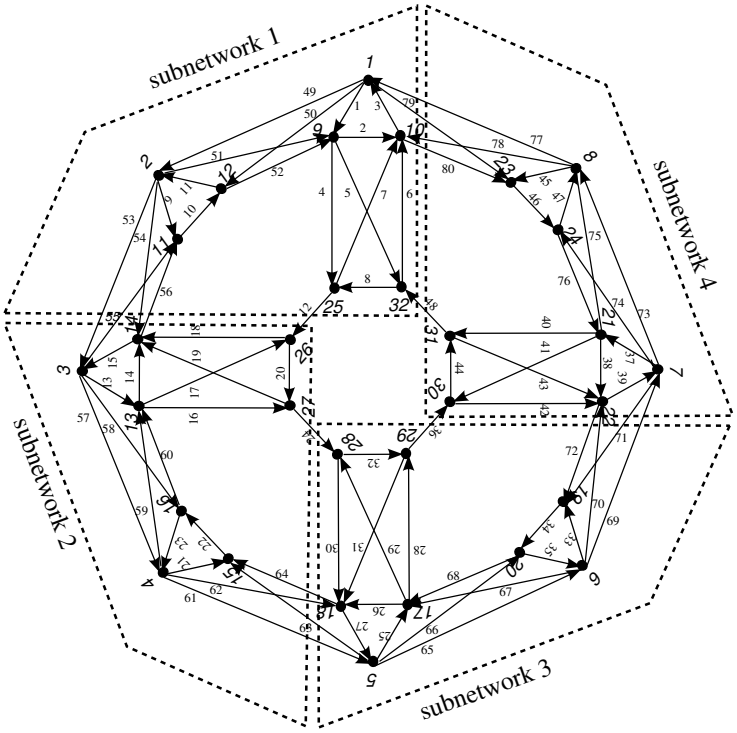

Figure 5.8: Division of the network used in the case study for dynamic traffic routing

representation is shown in Figure 5.9. For bi-level traffic routing based on merging nodes and links, we consider merging the nodes in the network into 9 aggregated nodes as shown in Figure 5.10. Correspondingly, the links in the network are merged into 24 aggregated links. The resulting aggregated network is shown in Figure 5.11.

We consider a case where there are 6 origin-destination pairs with nonzero traffic demand. More specifically, the 6 pairs of traffic demands are from node 1, node 2, node 3, node 3, node 4, and node 5 to node 6, node 6, node 6, node 8, node 8, and node 8, respectively. Besides, the profiles of the traffic demands are shown in Figure 5.12. The other parameters used in the simulation are: T = 10 s, k_{end} = 100, w_1 = 0.7, w_2 = 0.3, $J_{\text{TTS,typical}} = 1.0708 \times 10^6$ s, $J_{\text{TEC,typical}}$ = 918.06 kWh. In addition, the threshold to check convergence of solutions to traffic routing optimization problem used in Step iii) of the quasi-static procedure of Section 5.3.1 is set to 0.1, and the allowed maximum number of iterations used in Step iii) of the quasi-static procedure of Section 5.3.1 is set to 100.

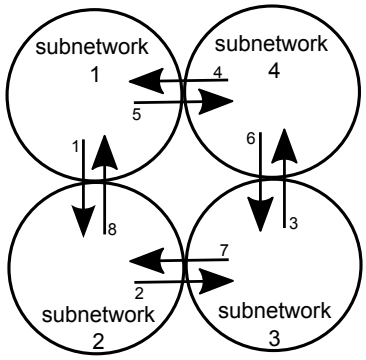

Figure 5.9: High-level representation of the network used in the case study for dynamic traffic routing

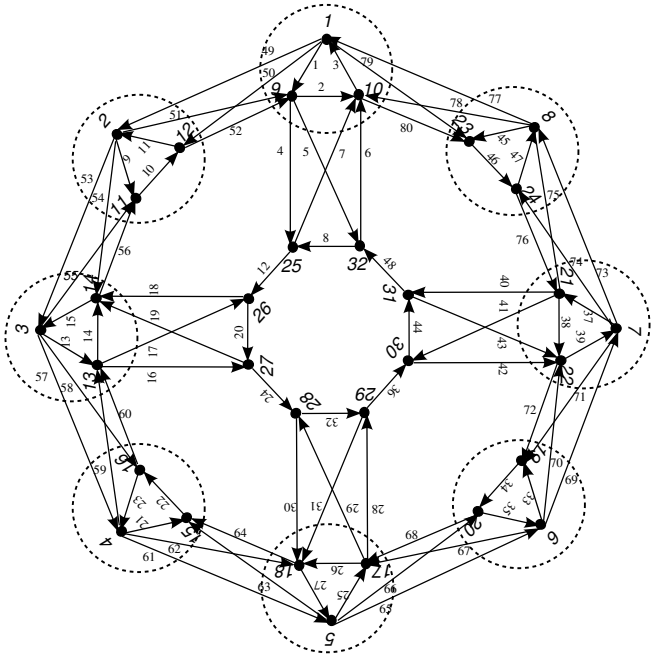

Figure 5.10: Merging nodes in the network used in the case study for dynamic traffic routing

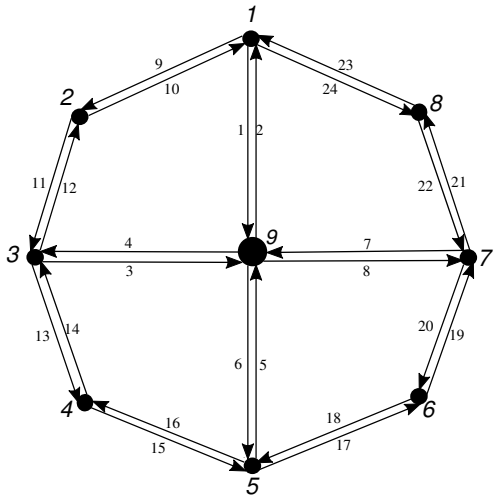

Figure 5.11: Aggregated representation of the network used in the case study for dynamic traffic routing

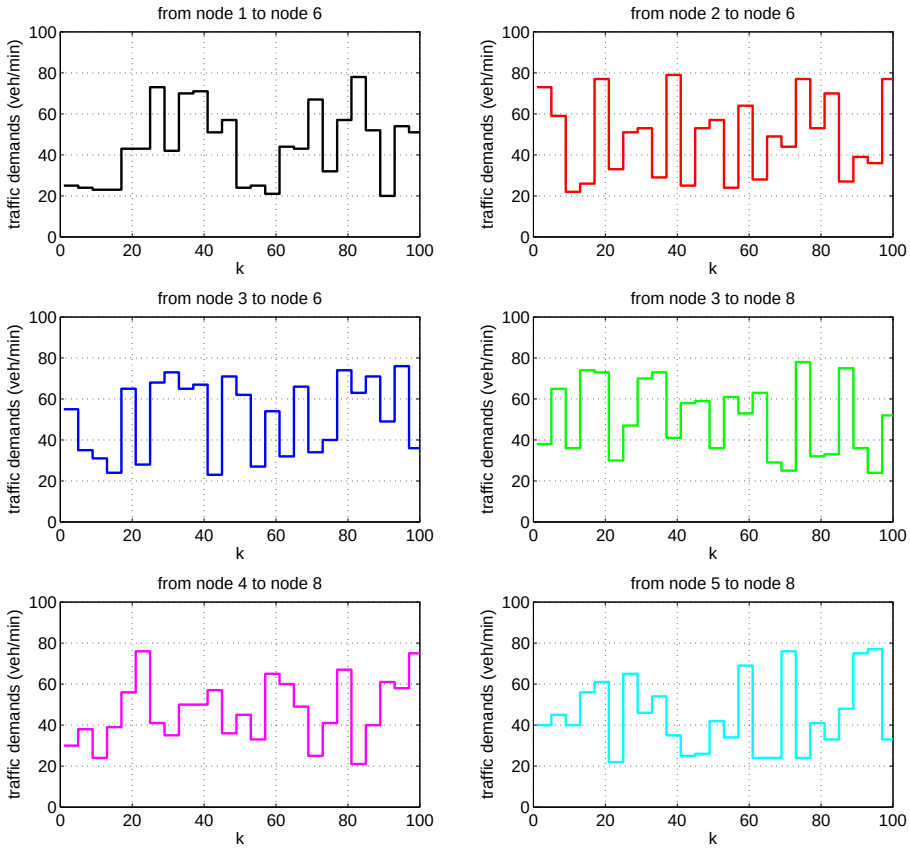

Figure 5.12: Profiles of traffic demands in the case study for dynamic traffic routing

Table 5.2: Simulation results for each subnetwork on the lower level

Subnetwork	$J_{\text{TTS},g}$ (s)	$J_{\text{TEC},g}$ (kWh)	J_g	computation time (s)
1	2.4944×10^5	346.06	0.2761	8.86
2	3.1938×10^5	379.66	0.3328	19.98
3	2.2491×10^5	249.97	0.2287	13.32
4	1.4000×10^5	194.30	0.1550	5.01

Moreover, for bi-level traffic routing based on merging nodes and links, at each iteration of the quasi-static procedure, the allowed maximum number of negotiations among agents of the distributed traffic routing algorithm of Section 5.4.3 is set to 100.

5.5.2 Hierarchical traffic routing

The simulation results of hierarchical traffic routing based on network division are as follows. At the high level, the quasi-static procedure stops at the 7th iteration. The resulting total time spent and total energy consumption of all high-level traffic flows and traffic queues are $J_{\text{TTS}} = 9.1940 \times 10^5$ s and $J_{\text{TEC}} = 852.18$ kWh, respectively. Besides, the value of the objective function is $J = 0.8930$. The computation time used in solving the high-level traffic routing problem is 0.73 s. At the low level, the quasi-static procedures for subnetworks 1, 2, 3 and 4 stop at the 9th, the 18th, the 15th, and the 7th iteration, respectively. Besides, for each subnetwork, the total time spent $J_{\text{TTS},g}$ and the total energy consumption $J_{\text{TEC},g}$ of traffic flows and traffic queues, the mismatch $J_{\text{FM},g}$ of high-level flows and lower-level flows, the value J_g of local objective function, and the computation time are summarized in Table 5.2. Finally, for the overall control performance evaluation, the resulting overall total time spent and the overall total energy consumption of traffic flows and traffic queues of the network are $J_{\text{overall,TTS}} = 7.8007 \times 10^5$ s and $J_{\text{overall,TEC}} = 1032.80$ kWh, respectively, and the overall performance $J_{\text{overall}} = 0.8474$.

5.5.3 Bi-level traffic routing using distributed control at the low level

For bi-level traffic routing based on merging nodes and links, we first used distributed traffic routing at the low level. Given the similarities between the urban transportation networks considered in this chapter and the transportation networks considered in [95], we adapt the distributed control scheme presented in [95] for distributing traffic routing at the low level. The simulation results are as follows. At the high level, the quasi-static procedure stops at the 21th iteration. The resulting total time spent and total energy consumption of all aggregated-level traffic flows and traffic queues are $J_{\text{TTS}} = 9.5690 \times 10^5$ s and $J_{\text{TEC}} = 792.52$ kWh, respectively. Besides, the value of the objective function is $J = 0.8845$. The computation time used in solving the traffic routing problem at the high level is 16.83 s.

At the low level, when the distributed control algorithm is used to solve the combined overall traffic routing problem, the local agents do not reach agreement on the interconnecting flows among aggregated nodes at all iterations in the quasi-static procedure. To explain this result, we take link 16 as an example. Link 16 is an interconnecting link between aggregated node 3 and aggregated node 9. The agents of

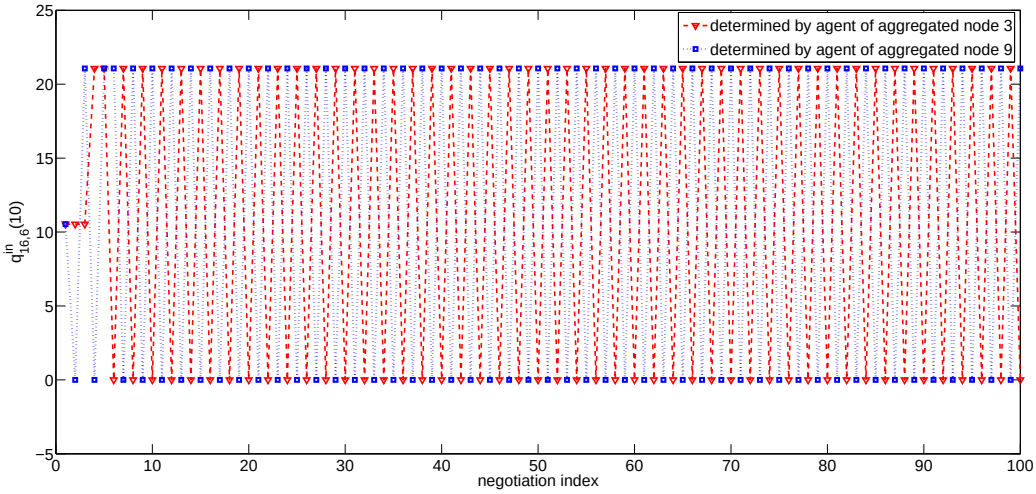

Figure 5.13: Values of $q^{\text{in}}_{16,6}(10)$ determined by the agent of aggregated node 3 and values of $q^{\text{in}}_{16,6}(10)$ determined by the agent of aggregated node 9 during negotiation at the 1st iteration of the quasi-static procedure. The dashed line and the dotted line are only used to highlight the oscillation of the values determined by the two agents.

aggregated node 3 and aggregated node 9 need to negotiate on the values of $q^{\text{in}}_{16,d}(k)$ for all d and k. However, the two agents do not reach agreement on those values during negotiation. More specifically, the values of $q^{\text{in}}_{16,6}(10)$ determined by the agent of aggregated node 3 and the values of $q^{\text{in}}_{16,6}(10)$ determined by the agent of aggregated node 9 during negotiation at the 1st iteration of the quasi-static procedure are shown in Figure 5.13. It is seen that the two agents have an oscillating disagreement on determining the values of $q^{\text{in}}_{16,6}(10)$ and such an oscillating disagreement will not be resolved by further negotiation. Actually, during negotiation, the two agents have an oscillating disagreement in determining the values of $q^{\text{in}}_{16,6}(k)$ for many simulation steps k. Moreover, the quasi-static procedure does not stop before the allowed maximum number of iterations was reached. Since the solutions to the overall combined traffic routing problem on the original level did not converge within the allowed maximum number of iterations, much computation time, which is actually 2.1467×10^5 s, had been spent by the time when the quasi-static procedure stopped.

Since further negotiation would not resolve the oscillating disagreement among agents, in order to save computation time, we reduced the allowed maximum number of negotiations among agents to 25. Then, the corresponding simulation results of solving the combined overall traffic routing problem using the distributed control algorithm are as follows. The computation time used in solving the combined overall traffic routing problem distributed control is 46013 s. Besides, for each aggregated node, the total time spent $J_{\text{TTS},\tilde{n}}$ and the total energy consumption $J_{\text{TEC},\tilde{n}}$ of traffic flows and traffic queues, and the value $J_{\tilde{n}}$ of the local objective function are summarized in Table 5.3. Finally, for the overall control performance evaluation, the resulting overall total time spent and the overall total energy consumption of traffic flows and traffic queues of the network are $J_{\text{overall,TTS}} = 1.1503 \times 10^6$ s and $J_{\text{overall,TEC}} = 945.87$ kWh, respectively, and the overall performance $J_{\text{overall}} = 1.0611$.

Table 5.3: Simulation results for each aggregated node with using distributed control at the low level

Aggregated node	$J_{\mathrm{TTS},\tilde{n}}$ (s)	$J_{\mathrm{TEC},\tilde{n}}$ (kWh)	$J_{\tilde{n}}$	computation time (s)
1	1.3973×10^5	144.89	0.1387	1986.75
2	6.1289×10^4	65.16	0.0614	1763.56
3	2.7192×10^5	193.54	0.2410	3734.21
4	6.6149×10^4	70.10	0.0662	1433.89
5	4.2476×10^5	353.15	0.3931	2157.33
6	1.1920×10^5	118.59	0.1167	1812.45
7	2.5838×10^5	213.05	0.2385	2311.72
8	1.0047×10^5	103.21	0.0994	3568.91
9	5.5771×10^5	401.10	0.4957	26145.13

Bi-level traffic routing using decentralized control at the low level

Alternatively, by fixing the values for the interconnecting variables by using (5.24), the overall combined control problem at the low level can be decomposed into a group of separated subproblems. Each subproblem can be solved individually by a local agent. In that case, an alternative decentralized control approach can be used. By using this alternative control approach, the computation time for solving the combined overall problem can be reduced. However, the quality of solution might be degraded.

In order to achieve a well-balanced trade-off between quality of solution and computation speed, we then solved the combined overall traffic routing problem at the low level using the alternative decentralized control approach. The simulation results of using the alternative decentralized control approach on the original level are as follows. The quasi-static procedures for aggreagted nodes 1, 2, 3, 4, 5, 6, 7, 8, and 9 stop at the 5th, the 5th, the 10th, the 4th, the 5th, the 5th, the 6th, the 9th, and the 28th iteration, respectively. Besides, for each aggregated node, the total time spent $J_{\mathrm{TTS},\tilde{n}}$ and the total energy consumption $J_{\mathrm{TEC},\tilde{n}}$ of traffic flows and traffic queues, the value $J_{\tilde{n}}$ of the local objective function, and the computation time are summarized in Table 5.4. Finally, for the overall control performance evaluation, the resulting overall total time spent and the overall total energy consumption of traffic flows and traffic queues of the network are $J_{\mathrm{overall,TTS}} = 1.0020 \times 10^6$ s and $J_{\mathrm{overall,TEC}} = 978.84$ kWh, respectively, and the overall performance $J_{\mathrm{overall}} = 0.9749$.

5.5.4 Comparison of the hierarchial approach and the bi-level approach

At the high level

The comparison of the hierarchical traffic routing approach and the bi-level traffic routing approach at the high level is summarized in Table 5.5. It is seen that the value of the objective function at the high level for the hierarchical approach is very close to that for the bi-level approach. Actually, the formulations of the high-level traffic routing problem in the two approaches are exactly the same. The only difference of the two approaches at this level is the representation of the network. More specifically, the numbers of nodes and links are

Table 5.4: Simulation results for each aggregated node with using decentralized control at the low level

Aggregated node	$J_{\mathrm{TTS},\tilde{n}}$ (s)	$J_{\mathrm{TEC},\tilde{n}}$ (kWh)	$J_{\tilde{n}}$	computation time (s)
1	1.8083×10^5	167.77	0.1730	5.45
2	8.8685×10^4	77.09	0.0832	3.76
3	2.1113×10^5	164.93	0.1919	9.70
4	8.7201×10^4	80.49	0.0833	2.88
5	2.6353×10^5	204.68	0.2382	5.02
6	1.5447×10^5	152.19	0.1507	3.33
7	1.4974×10^5	132.71	0.1413	6.64
8	1.6795×10^5	143.31	0.1564	6.04
9	4.3344×10^5	280.99	0.3752	56.27

Table 5.5: Comparison of the hierarchical approach and the bi-level approach at the high level

Approach	J_{TTS} (s)	J_{TEC} (kWh)	J	computation time (s)
Hierarchical	9.1940×10^5	852.18	0.8930	0.73
Bi-level	9.5690×10^5	792.52	0.8845	16.83

different in the two representations, and the lengths of the links in those two representations of the same network are also different. This implies that the travel costs of traffic demands from their origins to destinations would be similar in the two representations of the same network. Therefore, given the same problem formulation and the close representations of the same network, the value of the objective function at the high level for the hierarchical approach is close to that for the bi-level approach. However, the computation time used in solving the high-level traffic routing problem for the hierarchical approach is much shorter than the computation time used in solving the high-level traffic routing problem for the bi-level approach. That is because the number of aggregated nodes and aggregated links in Figure 5.11 is larger than the number of high-level nodes and high-level links in Figure 5.9. Thus, the number of decision variables of the aggregated-level traffic routing problem is larger than that of the high-level traffic routing problem. Moreover, in the quasi-static procedure, the number of iterations needed for solving the high-level traffic routing problem for the bi-level approach is larger than that needed for solving the high-level traffic routing problem for the hierarchical approach. Therefore, at the high level, the control approach based on network division is more efficient than the control approach based on merging nodes and links is solving the dynamic traffic routing problem.

At the low level

The comparison of the hierarchical traffic routing approach and the bi-level traffic routing approach at the low level is summarized in Table 5.6. Note that the computation time for each approach given in Table 5.6 is the parallel computation time, i.e. the computation time

Table 5.6: Comparison of the hierarchical approach and the bi-level approach at the low level

Approach	$J_{\text{overall,TTS}}$ (s)	$J_{\text{overall,TEC}}$ (kWh)	J_{overall}	computation time (s)
Hierarchical	7.8007×10^5	1032.80	0.8474	19.98
Bi-level distributed	1.1503×10^6	945.87	1.0611	26145.13
Bi-level decentralized	1.0020×10^6	978.84	0.9749	56.27

needed for solving the most time-consuming local subproblem. It is seen that hierarchical approach at the low level performs better than the two variants of the bi-level approach. More specifically, the overall J_{overall} and the computation time of the hierarchical approach is the lowest among the three approaches.

Besides, for the two variants of the bi-level approach, the one using decentralized control performs better than the one using distributed control in both overall performance J_{overall} and computation time. The results can be explained as follows. In the distributed traffic routing control of the bi-level approach, the agents do not reach agreement on the interconnecting flows. By the time when the allowed maximum number of negotiation is reached, the agents just take their current decisions, which are parts of the oscillating disagreements among the agents and can be far away from the optimum. As a result, the overall performance of all agents may be far away from the global optimal performance. Besides, since the local agents do not reach agreement on the interconnecting flows at all iterations in the quasi-static procedure and the quasi-static procedure does not stop until the allowed maximum number of iterations is reached, much computation time is spent on the unrewarding negotiation among agents and the non-converging quasi-static procedure. In decentralized traffic routing control of the bi-level approach, the flows on the interconnecting links are first fixed by using (24) and then each subproblem is solved individually. Since the fixed values of traffic flows on the interconnecting links are not optimal, the resulting overall performance is also worse than the global overall optimal performance. Moreover, the computation time for solving the local traffic routing problem for aggregated node 9, which is the most time-consuming node in the bi-level approach, is longer than that for solving the local traffic routing problem for subnetwork 2, which is the most time-consuming subnetowrk in the hierarchical approach.

5.6 Summary

In this chapter we have addressed the dynamic traffic routing problem for urban transportation networks. In general, this problem is computationally very hard to solve. To reduce the computational efforts in solving the problem, we have proposed a novel hierarchical control approach based on network division and a novel bi-level control approach based on merging nodes and links. Simulation-based case studies show that the hierarchical control approach based on network division is more effective than the bi-level control approach based on merging nodes and links in solving the dynamic traffic routing problem.

One possible extension of the work presented in this chapter is to perform more detailed case stuidies for an extensive assessment of the efficiency of the hierarchical traffic

routing approach and the bi-level traffic routing approach, e.g., quantifying the communication efforts of the two approaches.

Chapter 6

Co-optimization of the Orientation of Road Sections and the Routes of Traffic Flows

In this chapter we address the co-optimization problem that jointly determines the orientation of road sections and routes of traffic flows by assuming that the orientation of each road section in the network can be changed in each control period. We consider a circular orientation for each elementary cycle, which is a series of connected road sections encircling an area that is not encircled by other string of roads in the network, and we map the orientations of road sections using the orientation of these elementary cycles. Given the number of elementary cycles is much smaller than the number of road sections in the network, the number of binary variables involved in the co-optimization problem is reduced substantially w.r.t considering the orientation of each road section independently. Therefore, the resulting co-optimization problem can be solved more efficiently.

Parts of this chapter have been included in [80].

6.1 Introduction

Due to inadequate urban planning, in many big cities, the main commercial and business areas are concentrated in the center while the residential areas are spread outside the city center. As people commute between their work places and their homes daily, in those cities there are always high traffic demands from outside the city center to the city center in the morning rush hours as well as high traffic demands from the city center to outside the city center in the evening rush hours. Conventionally, after the urban transportation networks are constructed, the orientation of the road sections in the network is fixed. As a consequence, the roads directed from city center to outside the city center are not sufficiently used in the morning rush hours. A similar argument holds for the roads directed from outside the city center to the city center in the evening rush hours. Therefore, in this chapter we consider co-optimization of the orientation of road sections and the routes of traffic flows for urban transportation networks.

Different from the urban transportation network design problems considered in [46], where the orientation of roads in the network are fixed once determined, we assume the orientation of each road section in an urban transportation network can be changed in each

control period, e.g., in the morning rush hours and in the evening rush hours. We focus on the co-optimization problem that jointly determines the orientation of road sections in the network and the routes of traffic flows. Generally, the co-optimization problem for a large-scale network contains a large number of binary optimization variables, which makes the problem computationally hard to solve. However, by considering a circular orientation[1] in each elementary cycle in the graph corresponding to the road network and mapping the orientations of road sections using the orientation of these elementary cycles, the number of binary variables involved in the co-optimization problem is reduced substantially w.r.t considering the orientation of each road section independently.

With respect to the literature, the main contribution of this chapter consists in proposing a novel mixed-integer linear programming formulation with a reduced number of binary optimization variables for the co-optimization problem.

The chapter is organized as follows. Section 6.2 describes the general co-optimization problem. In Section 6.3, we define the variables used in the chapter. After that, the relation between the orientation of links and the circular orientation of traffic flows in each elementary cycle is described in Section 6.4. Besides, the model of dynamics of traffic and the objective function of the co-optimization problem are presented in Section 6.5 and in Section 6.6, respectively. Further, the co-optimization problem is formulated in Section 6.7. Moreover, Section 6.8 illustrates a simulation case study of the proposed method. Finally, in Section 6.9, we summarize the main contribution of the chapter and propose some ideas for future work.

6.2 Problem description

Conceptually, an urban road network can be represented as a graph shown in Figure 6.1, where each elementary cycle has been indicated by a dashed line. We assume that each road section that is shared by two elementary cycles, has two independent unidirectional lanes. Each unidirectional lane in a road section is referred to as a link and each intersection is referred to as a node. Note that efficient algorithms for detecting elementary cycles in a given network have been developed in [72, 122]. In our work, we assume a network is given with elementary cycles already determined, and we only focus on the co-optimization problem of determining the orientation of road sections and the routes of traffic flows.

We assume that each link has a constant travel time that is determined statistically based on the historical data. We also assume that all O-D (origin-destination) traffic demands are given by higher-level controllers. Provided the O-D demands, a model describing the dynamics of traffic flows in the network is given in Section 6.5. We aim to jointly determine the orientation of the elementary cycles and the optimal routes for traffic flows (i.e., the optimal assignment of traffic flows on each link) so that all O-D traffic demands are handled and the total travel cost including e.g. the total time spent and the total energy consumption of all the traffic flows is minimized.

[1] In many urban areas, one-way road sections are used to achieve higher traffic flow as drivers may avoid encountering oncoming traffic or turning through oncoming traffic. This one-way scheme typically results in (elementary) cycles with a circular orientation.

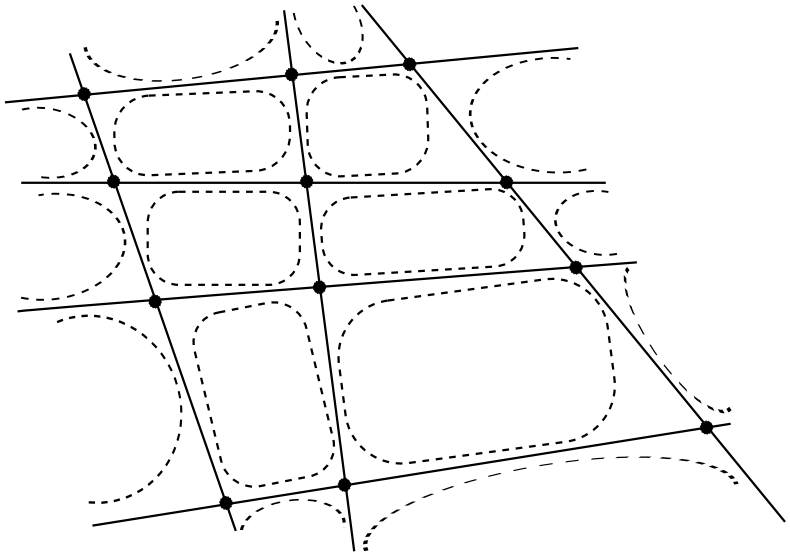

Figure 6.1: Schematic representation of an urban road network with elementary cycles (dashed lines)

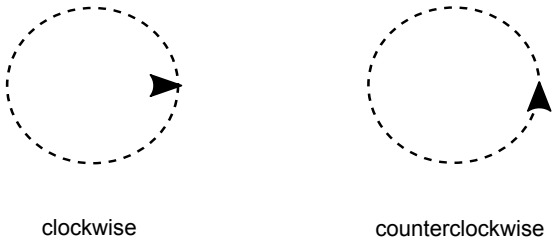

Figure 6.2: Two possible circular orientations for traffic flows in a ring of roads

6.3 Definitions

We consider there are two possible circular orientations, i.e. clockwise and counterclockwise, for traffic flows in each elementary cycle, as shown in Figure 6.2. After that, we define

- k: simulation step counter
- T: length of the simulation time interval
- t_l: average travel time on link l
- e_l: average energy consumption of a vehicle traveling through link l
- $\mathscr{N}$: set of nodes
- $\mathscr{L}$: set of links
- $\mathscr{D}$: set of final destinations
- $\mathscr{I}_n$: set of incoming links of node n
- $\mathscr{O}_n$: set of outgoing links of node n
- $D_{n,d}(k)$: traffic demand from node n to final destination d for interval $\left[kT,(k+1)T\right)$

Figure 6.3: Definition of $\sigma_{l,n}$

- $q_{l,d}^{\text{in}}(k)$: flow entering link l traveling towards final destination d for interval $\left[kT,(k+1)T\right)$
- $q_{l,d}^{\text{out}}(k)$: flow leaving link l traveling towards final destination d for interval $\left[kT,(k+1)T\right)$
- $Q_{n,d}(k)$: queue length of traffic towards final destination d but waiting at node n for interval $\left[kT,(k+1)T\right)$
- q_l^{cap}: capacity of link l
- δ_i: circular orientation of traffic flows in elementary cycle i; $\delta_i = 1$ represents clockwise orientation while $\delta_i = 0$ represents counterclockwise orientation.
- $\sigma_{l,n}$: direction of link l that is connected to node n. As illustrated in Figure 6.3, $\sigma_{l,n} = 1$ indicates that link l is directed towards node n, and $\sigma_{l,n} = 0$ indicates that link l is directed away from node n.
- $\mathscr{L}_n$: set of incoming links and outgoing links of node n
- $\mathscr{N}_i$: set of nodes in elementary cycle i
- $\mathscr{R}_i$: set of links in elementary cycle i

6.4 Model of the relation between the orientation of links and the circular orientation in each elementary cycle

The clockwise (or counterclockwise) orientation of traffic flows in an elementary cycle can be mapped to the orientation of the links in that elementary cycle. More specifically, given δ_i for each elementary cycle i, the variables[2] $\sigma_{l,n}$ for all $n \in \mathscr{N}_i$ and $l \in \mathscr{R}_i \cap \mathscr{L}_n$ are explicitly determined by δ_i as follows. First, the orientation of elementary cycle i is mapped to the orientation of link l. Next, given the orientation of link l, the variable $\sigma_{l,n}$ is either given by

$$\sigma_{l,n} = \delta_i \tag{6.1}$$

as the cases (a) and (b) indicated in Figure 6.4, or is given by

$$\sigma_{l,n} = 1 - \delta_i \tag{6.2}$$

as the cases (c) and (d) indicated in Figure 6.4.

[2]Note that at first sight a single variable σ_l would seem sufficient to describe the direction of the traffic flow on link l. However, this would make the equation of the traffic model in Section 6.5 much more complex. Therefore, we also include the node index as an extra subscript in σ.

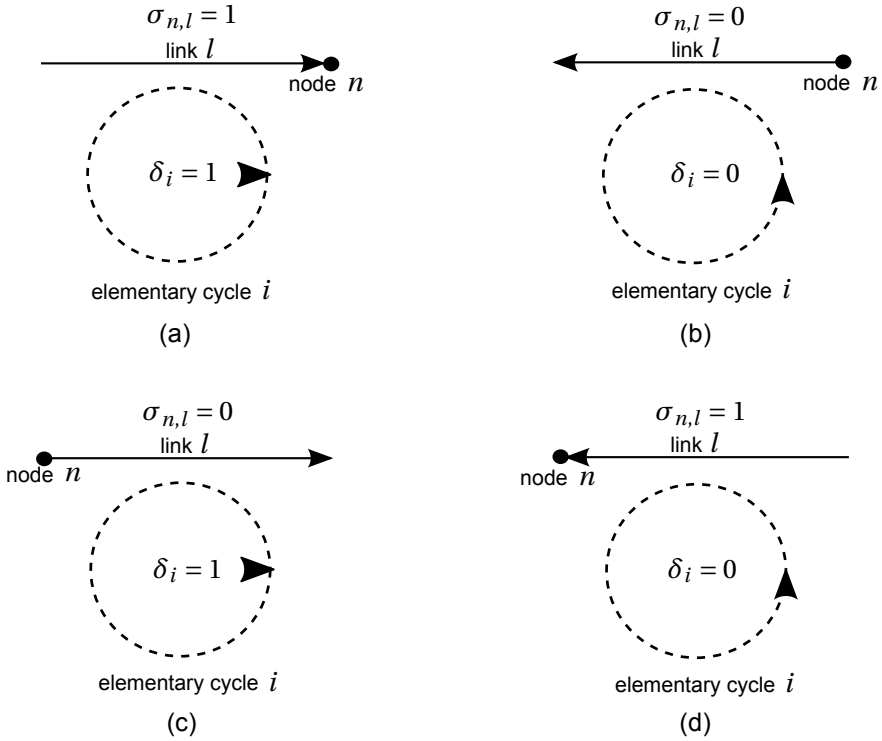

Figure 6.4: Relation between the circular orientation of an elementary cycle and the orientation of the links in that elementary cycle.

6.5 Model of dynamics of traffic flows

For modeling the dynamics of traffic flows, we assume that the average travel time t_l of link l is an integer multiple of T. Based on the circular orientation of each elementary cycle, we describe the model of the dynamics of traffic flows considering flow consistency at each node, time delay between inflow and outflow on each link, non-negativity of traffic flows and traffic queues, and capacity of each link.

Firstly, at each node n, the sum of outgoing traffic flows from node n is equal to the sum of traffic flows entering node n. More specifically, considering the orientation of the links that are connected to node n, the flow consistency at each node n is given by

$$\sum_{l \in \mathscr{L}_n} q^{\text{in}}_{l,d}(k) \cdot (1 - \sigma_{l,n}) \cdot T + Q_{n,d}(k) - Q_{n,d}(k-1) = D_{n,d}(k) \cdot T + \sum_{l \in \mathscr{L}_n} q^{\text{out}}_{l,d}(k) \cdot \sigma_{l,n} \cdot T \qquad (6.3)$$

$$\forall d \in \mathscr{D},\ n \neq d$$

Note that if $\sigma_{l,n} = 1$ with $l \in \mathscr{L}_n$, then link l is an incoming link of node n. Accordingly, $q^{\text{out}}_{l,d}(k) \cdot \sigma_{l,n} \cdot T$ is equal to the incoming traffic flow of node n from link l at time step k, and $q^{\text{in}}_{l,d}(k) \cdot (1 - \sigma_{l,n}) \cdot T$ is equal to zero. If $\sigma_{l,n} = 0$ with $l \in \mathscr{L}_n$, then link l is an outgoing link of node n. Accordingly, $q^{\text{in}}_{l,d}(k) \cdot (1 - \sigma_{l,n}) \cdot T$ is equal to the outgoing traffic flow from node n that enters link l at time step k, and $q^{\text{out}}_{l,d}(k) \cdot \sigma_{l,n} \cdot T$ is equal to zero. Therefore, $\sum_{l \in \mathscr{L}_n} q^{\text{in}}_{l,d}(k) \cdot (1 - \sigma_{l,n}) \cdot T$ is equal to the total traffic flow leaving node n through its outgoing links at time step k, and $\sum_{l \in \mathscr{L}_n} q^{\text{out}}_{l,d}(k) \cdot \sigma_{l,n} \cdot T$ is equal to the total traffic flow entering node n through its incoming links at time step k. Besides, $D_{n,d}(k) \cdot T$ is equal to the amount of traffic flow entering node n at time step k due to the traffic demand, and $Q_{n,d}(k) - Q_{n,d}(k-1)$ is the

difference in queue length at node n.

Secondly, the relationship between the incoming flow and the outgoing flow of each link l is given by

$$q_{l,d}^{\text{out}}(k) = q_{l,d}^{\text{in}}\left(k - \frac{t_l}{T}\right), \quad \forall d \in \mathscr{D} \tag{6.4}$$

Thirdly, since the traffic flows must not be negative and the sum of traffic flows on a link going to different destinations must not exceed the capacity of the link, the flow constraints are given by

$$q_{l,d}^{\text{in}}(k) \geq 0, \quad \forall l \in \mathscr{L},\ \forall d \in \mathscr{D} \tag{6.5}$$

$$\sum_{d \in \mathscr{D}} q_{l,d}^{\text{in}}(k) \leq q_l^{\text{cap}}, \quad \forall l \in \mathscr{L} \tag{6.6}$$

Finally, the queue length of traffics waiting at a node must not be negative. Therefore, we have

$$Q_{n,d}(k) \geq 0, \quad \forall n \in \mathscr{N},\ \forall d \in \mathscr{D} \tag{6.7}$$

6.6 Objectives

We consider minimizing the total travel cost including the total time spent and the total energy consumption of all traffic flows and traffic queues. More specifically, the total time spent of all traffic flows and traffic queues is given by

$$\begin{aligned} J_{\text{TTS}} = &\sum_{d \in \mathscr{D}} \sum_{l \in \mathscr{L}} \sum_{k=0}^{k_{\text{end}}-1} q_{l,d}^{\text{in}}(k) \cdot T \cdot t_l + \sum_{d \in \mathscr{D}} \sum_{n \in \mathscr{N}} \sum_{k=0}^{k_{\text{end}}-1} Q_{n,d}(k) \cdot T \\ &+ \sum_{d \in \mathscr{D}} \sum_{l \in \mathscr{L}} q_{l,d}^{\text{in}}(k_{\text{end}}) \cdot T \cdot \tau_{l,d} + \sum_{d \in \mathscr{D}} \sum_{n \in \mathscr{N}} Q_{n,d}(k_{\text{end}}) \cdot \iota_{n,d} \end{aligned} \tag{6.8}$$

where $\tau_{l,d}$ denotes a measure of the remaining average travel time from link l to destination d, and $\iota_{n,d}$ denotes a measure of the remaining travel time from node n to destination d. One possible way to determine $\tau_{l,d}$ and $\iota_{n,d}$ is using t_l for all links and the shortest time routes computed by *Dijkstra's algorithm* [40]. The total energy consumption of all traffic flows and traffic queues is given by

$$\begin{aligned} J_{\text{TEC}} = &\sum_{d \in \mathscr{D}} \sum_{l \in \mathscr{L}} \sum_{k=0}^{k_{\text{end}}-1} q_{l,d}^{\text{in}}(k) \cdot T \cdot e_l + \sum_{d \in \mathscr{D}} \sum_{n \in \mathscr{N}} \sum_{k=0}^{k_{\text{end}}-1} Q_{n,d}(k) \cdot e_{\text{idle},T} \\ &+ \sum_{d \in \mathscr{D}} \sum_{l \in \mathscr{L}} q_{l,d}^{\text{in}}(k_{\text{end}}) \cdot T \cdot \chi_{l,d} + \sum_{d \in \mathscr{D}} \sum_{n \in \mathscr{N}} Q_{n,d}(k_{\text{end}}) \cdot \varsigma_{n,d} \end{aligned} \tag{6.9}$$

where $e_{\text{idle},T}$ denotes the average amount of energy consumed by a vehicle idling for the interval of length T, $\chi_{l,d}$ denotes a measure of the remaining energy consumption of a vehicle from link l to destination d, and $\varsigma_{n,d}$ denotes a measure of the remaining energy consumption of a vehicle from node n to destination d. One possible way to determine $\chi_{l,d}$ and $\varsigma_{n,d}$ is using e_l for all links and the shortest time routes.

Note that (6.8) and (6.9) are linear expressions of $q_{l,d}^{\text{in}}(k)$ and $Q_{n,d}(k)$ for all $l \in \mathscr{L}$, $n \in \mathscr{N}$,

and $d \in \mathscr{D}$.

6.7 Problem formulation

Note that $\sigma_{l,n}$ for all links and nodes are just introduced to help describe the modeling of dynamics of traffic flows. In Section 6.4, we have already shown that $\sigma_{l,n}$ for all $n \in \mathscr{N}_i$ and $l \in \mathscr{R}_i \cap \mathscr{L}_n$ can be explicitly expressed by using δ_i. Therefore, $\sigma_{l,n}$ in (6.3) for all links and nodes can be replaced by δ_i for all elementary cycles. Since the number of elementary cycles is much smaller than the number of links in the network, the number of binary optimization variables of the problem is substantially reduced by replacing all $\sigma_{l,n}$ with all δ_i.

Although the number of binary optimization variables is then greatly reduced, the problem is still very hard to solve due to the nonlinearities of $q^{\text{in}}_{l,d}(k) \cdot \delta_i$ and $q^{\text{out}}_{l,d}(k) \cdot \delta_i$ in (6.3). However, according to [11], (6.3) can be transformed into a set of mixed integer linear expressions by introducing auxiliary real variables $z^{\text{in}}_{l,d}(k) = q^{\text{in}}_{l,d}(k) \cdot \delta_i$, $z^{\text{out}}_{l,d}(k) = q^{\text{out}}_{l,d}(k) \cdot \delta_i$ and the following extra linear constraints:

$$
\begin{aligned}
&z^{\text{in}}_{l,d}(k) \le M^{\text{in}}_{l,d} \cdot \delta_i \\
&z^{\text{in}}_{l,d}(k) \ge m^{\text{in}}_{l,d} \cdot \delta_i \\
&z^{\text{in}}_{l,d}(k) \le q^{\text{in}}_{l,d}(k) - m^{\text{in}}_{l,d} \cdot (1-\delta_i) \\
&z^{\text{in}}_{l,d}(k) \ge q^{\text{in}}_{l,d}(k) - M^{\text{in}}_{l,d} \cdot (1-\delta_i) \\
&z^{\text{out}}_{l,d}(k) \le M^{\text{out}}_{l,d} \cdot \delta_i \\
&z^{\text{out}}_{l,d}(k) \ge m^{\text{out}}_{l,d} \cdot \delta_i \\
&z^{\text{out}}_{l,d}(k) \le q^{\text{out}}_{l,d}(k) - m^{\text{out}}_{l,d} \cdot (1-\delta_i) \\
&z^{\text{out}}_{l,d}(k) \ge q^{\text{out}}_{l,d}(k) - M^{\text{out}}_{l,d} \cdot (1-\delta_i)
\end{aligned}
$$

where $M^{\text{in}}_{l,d}$ and $m^{\text{in}}_{l,d}$ are respectively the upper bound and lower bound of $q^{\text{in}}_{l,d}(k)$, and $M^{\text{out}}_{l,d}$ and $m^{\text{out}}_{l,d}$ are respectively the upper bound and lower bound of $q^{\text{out}}_{l,d}(k)$. Therefore, the co-optimization problem can be formulated as a mixed-integer linear programming (MILP) problem with binary decision variables δ_i for all elementary cycles of roads, and real decision variables $q^{\text{in}}_{l,d}(k)$, $Q_{n,d}(k)$ and $z^{\text{in}}_{l,d}(k)$ for $l \in \mathscr{L}$, $n \in \mathscr{N}$, and $d \in \mathscr{D}$. More specifically, the co-optimization problem of jointly determining the circular orientation of road sections and routes of traffic flows is formulated by

$$
\begin{aligned}
\underset{\boldsymbol{q}^{\text{in}},\boldsymbol{q}^{\text{out}},\boldsymbol{Q},\boldsymbol{z}^{\text{in}},\boldsymbol{z}^{\text{out}},\boldsymbol{\delta}}{\text{minimize}} \quad & J := w_1 \frac{J_{\text{TTS}}}{J_{\text{TTS,typical}}} + w_2 \frac{J_{\text{TEC}}}{J_{\text{TEC,typical}}} \\
\text{subject to} \quad & \text{flow dynamics over the simulated period} \\
& \text{initial conditions}
\end{aligned}
\tag{6.10}
$$

where $\boldsymbol{q}^{\text{in}}$, $\boldsymbol{q}^{\text{out}}$, $\boldsymbol{Q}$, $\boldsymbol{z}^{\text{in}}$ and $\boldsymbol{z}^{\text{out}}$ are respectively the compact expressions containing $q^{\text{in}}_{l,d}(k)$, $q^{\text{out}}_{l,d}(k)$, $Q_{n,d}(k)$, $z^{\text{in}}_{l,d}(k)$ and $z^{\text{out}}_{l,d}(k)$ for all $l \in \mathscr{L}$, $d \in \mathscr{D}$ and $k \in \{0,1,2,...,k_{\text{end}}-1\}$, and $\boldsymbol{\delta}$ is the compact expression containing δ_i for all elementary cycles of roads. Besides, $J_{\text{TTS,typical}}$ and $J_{\text{TEC,typical}}$ denotes typical values of the total time spent and the total energy consumption of traffic flows over the simulated period. Note that problem (6.10) can be solved efficiently by using state-of-the-art MILP solvers like CPLEX, GUROBI, MOSEK, or XPRESS.

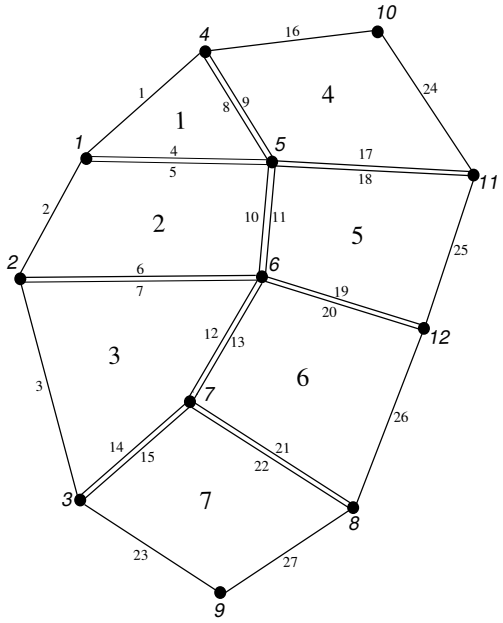

Figure 6.5: The network used in the case study for the co-optimization of the orientation of road sections and the routes of traffic flows

Table 6.1: Lengths of links in the network shown in Figure 6.5

link	length (m)	link	length (m)	link	length (m)
1	400	10, 11	300	21, 22	600
2	400	12, 13	400	23	400
3	700	14, 15	400	24	500
4, 5	450	16	450	25	500
6, 7	700	17, 18	600	26	650
8, 9	300	19, 20	500	27	400

6.8 Simulation study

For co-optimization of the orientation of road sections and the routes of traffic flows, we consider the network shown in Figure 6.5, where there are 12 nodes, 27 links, and 7 elementary cycles. The lengths of all links are listed in Table 6.1. We consider a case where there are 4 origin-destination pairs with nonzero traffic demands. More specifically, the 4 pairs of traffic demands are from node 1, node 2, node 8, and node 11 to node 9, node 10, node 10, and node 2, respectively. The profiles of the traffic demands are shown as in Figure 6.6. The other parameters used in the simulation are: $T = 10$ s, $k_{\text{end}} = 100$, $w_1 = 0.7$, $w_2 = 0.3$, $J_{\text{TTS,typical}} = 5.0472 \times 10^5$ s, and $J_{\text{TEC,typical}} = 155.76$ kWh. The simulation is performed using Matlab 2015b on a cluster computer consisting of 4 blades with 2 eight-core E5-2643 processors, and 3.3 GHz clock rate and 64 GiB memory per blade.

In the simulation, we solve the resulting mixed-integer linear programming problem (6.10) using CPLEX. The computation time needed for solving the problem is 33.32 s. The results are that the links in elementary cycles 1, 2, 3, and 7 are directed to circle counterclockwise while the links in elementary cycles 4, 5, and 6 are directed to circle

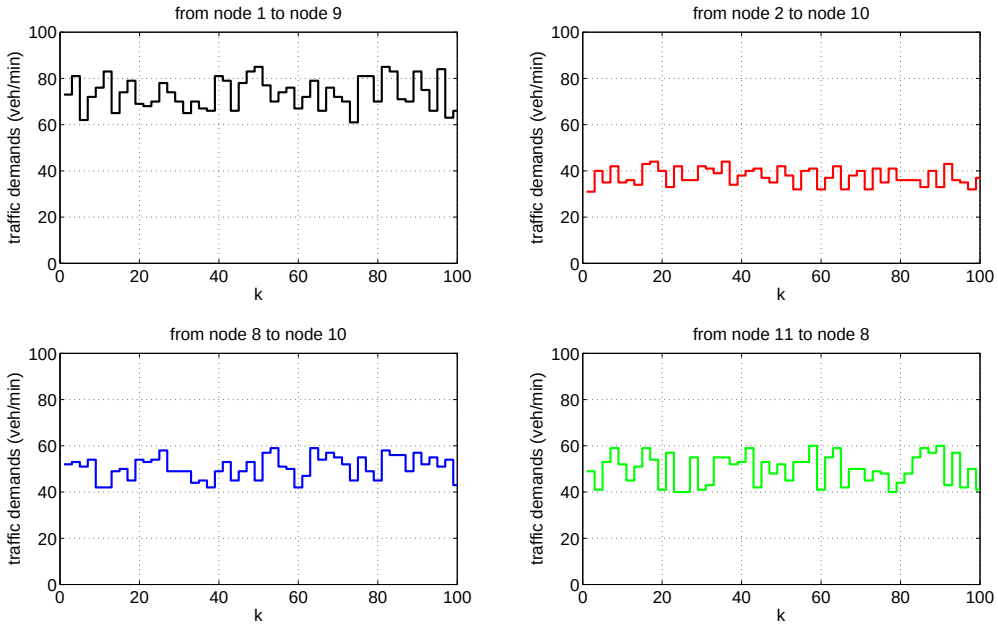

Figure 6.6: Profiles of traffic demands in the case study for the co-optimization of the orientation of road sections and routes of traffic flows

clockwise. The resulting network with the orientation of each link indicated is shown in Figure 6.7. Besides, the traffic demands from node 1 to node 9 take the route consisting of links 2, 3, and 23 in sequence; the traffic demands from node 11 to node 8 take the route consisting of links 25 and 26 in sequence. Moreover, the traffic demands from node 2 to node 10 first take link 6 to reach node 6 and the traffic demands from node 8 to node 10 first take link 21 and link 12 to reach node 6. Afterwards, the two traffic demands mix together and take either link 10 or link 11 to reach node 5 and further take either link 8 or link 9 to reach node 4. Finally, from node 4, the traffic flows take link 16 to reach their final destination, i.e., node 10. The resulting total time spent and the total energy consumption of all traffic flows and all traffic queues are $J_{\mathrm{TTS}} = 3.8685 \times 10^5$ s and $J_{\mathrm{TEC}} = 164.45$ kWh, respectively, and the value of the objective function is $J = 0.8533$.

The simulation results are explained as follows. If the links in elementary cycle 1 and elementary cycle 2 were directed clockwise, the least costly route for the traffic demands from node 2 to node 10 would be the one consisting of links 2, 1, and 16. However, in that case, the traffic demands from node 1 to node 9 could not take their least costly route consisting of links 2, 3, and 23. Given the profiles of traffic demands shown in Figure 6.6, the traffic demand from node 2 to node 10 is about half as many as those from node 1 to node 9. Therefore, from the system optimum point of view, it is more beneficial to enable the traffic demand from node 1 to node 9 to take the least costly route, which is only possible if the links in elementary cycles 2, 3, and 7 are directed counterclockwise. Given the links in elementary cycle 2 are directed counterclockwise, the traffic demand from node 2 to node 10 cannot take the least costly route. In that case, the best alternative is to take link 6 to reach node 6 and then take the least costly route from there to node 10. Actually, the least costly route from node 6 to node 10 is the shortest route consisting of either link 10 or link 11 from node 6 to node 5, followed by either link 8 or link 9 from node 5 to node 7, and then finally link 16 from node 4 to node 10. Therefore, for the whole network, it is better to let the

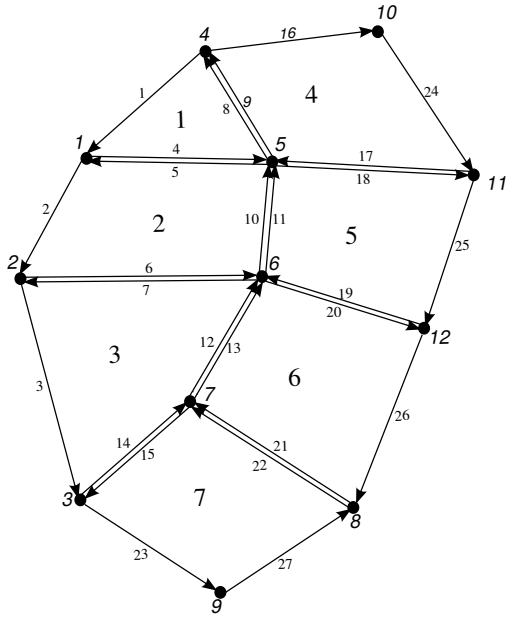

Figure 6.7: The solution for orientation of the network in the case study for the co-optimization of the orientation of road sections and routes of traffic flows

links in elementary cycle 4 to be directed clockwise. Likewise, the least costly route of the traffic demand from node 8 to node 10 conflicts with that of the traffic demand from node 11 to node 8. Although the traffic demand from node 8 to node 10 is comparable to those from node 11 to node 8, the extra cost for the traffic demand from node 11 to node 8 to take an alternative route is much higher than that for the traffic demand from node 8 to node 10 to take its alternative route. Therefore, for the whole network, it is more beneficial to let the traffic demand from node 11 to node 8 to take the least costly route consisting of links 25 and 26, which is only possible if the links in elementary cycle 5 and elementary cycle 6 are directed clockwise. After that, the best alternative for the traffic demand from node 8 to node 10 is to take link 21 to reach node 7, then to take link 12 or link 13 to reach node 6, and to take the least costly route from node 6 to node 10.

6.9 Summary

We have addressed the co-optimization problem that jointly determines the orientation of road sections and the routes of traffic flows for urban transportation networks. To reduce the computational efforts for solving the problem, we have started from a setting in which the orientation of traffic flows is circular in each elementary cycle in the network. The resulting co-optimization problem can be solved more efficiently than considering the orientation of each road section independently. A simulation-based case study shows that the novel approach is suitable for on-line optimization. Possible extensions of the work presented in this chapter include combining the dynamic orientating of road sections and the dynamic routing of traffic flows i.e. considering dynamic travel time of each link in the network, and proposing an efficient solution method for the whole problem.

Part II

Multi-agent control of hybrid systems

Chapter 7

Multi-Agent Model Predictive Control of Hybrid Systems with Limited Information Sharing

In this chapter we develop a multi-agent model predictive control method for a class of hybrid systems governed by discrete inputs and subject to global hard constraints. We assume that for each subsystem the local objective function is convex and the local constraint function is strictly increasing with respect to the local control variable. The proposed multi-agent control method is based on a distributed resource allocation coordination algorithm and it only requires limited information sharing among the local agents of the subsystems. Thanks to primal decomposition of the global constraints, the distributed algorithm can always guarantee global feasibility of the local control decisions, even in the case of premature termination. Moreover, since the control variables are discrete, a mechanism is developed to branch the overall solution space based on the outcome of the resource allocation coordination algorithm at each node of the search tree. Finally, the proposed multi-agent control method is applied to the charging control problem of electric vehicles under constrained grid conditions. This case study highlights the effectiveness of the proposed method.

Parts of this chapter have been included in [79].

7.1 Introduction

7.1.1 Multi-agent hybrid systems and their control

Multi-agent systems, like transportation systems, manufacturing systems, power systems, financial systems, are composed of multiple subsystems with interactions [64]. Multi-agent systems research is facing a variety of challenges [71], of which a crucial one is to design mechanisms for coordinating agents that have limited information sharing with each other in order to protect confidential information of local subsystems while at the same time still aiming for global performance [43]. Typical global control goals for multi-agent systems involve synchronizing motions of agents, maximizing resource utility, and minimizing control costs. In order to achieve globally satisfactory performance, given limited information of other subsystems, the agents need to assist each other to make better decisions about their actions [24]. Thanks to its straightforward design procedure, where

hard system constraints are incorporated directly as inequalities in the formulation of the control problem, model predictive control has shown to be a promising control strategy for multi-agent systems [23, 42, 95, 112].

However, the cooperation among agents is made much more difficult when the individual agents have to regulate hybrid subsystems [27] that contain both continuous components and discrete components, such as switches and overrides. In fact, this will result in having to solve mixed-integer programming problems in a distributed way, for which there has not yet been a universally successful algorithm [49]. Moreover, many system-theoretic concepts and control strategies, such as model predictive control and Artificial Intelligence based control [21, 96], still require further examination and research in this setting [91].

7.1.2 Multi-agent model predictive control for hybrid systems with global hard constraints

We focus on a class of hybrid systems that are governed by discrete inputs and that are subject to global hard constraints. In particular, each subsystem is characterized by a convex objective function and a strictly increasing constraint function with respect to the local control variable. Besides, each subsystem only shares limited information with the external environment. Actually, such hybrid systems are ubiquitous, and an important example are a group of systems with on/off switches sharing a given amount of resources. More specifically, concrete real-life examples include the charging of a fleet of electric vehicles sharing a given power level provided by the grid, and the operation of a number of appliances sharing a given amount of energy in smart buildings. We aim to develop a multi-agent model predictive control method for such a class of hybrid systems based on a distributed resource allocation coordination algorithm.

A resource allocation is a plan for using the available resources to achieve goals for the future. In principle, such planning may be done by centrally scheduling the actions of the systems that require resources [59]. However, for reasons of scalability and fast computation, it will not be tractable to schedule the actions of a large number of systems in a centralized way. Actually, the scheduling of the actions of the systems that require resources can be done in a distributed way based on the primal decomposition [20] of the overall problem. More specifically, in primal decomposition, which is naturally applicable to resource allocation scenarios, the allocation of resources can be represented by auxiliary variables and these variables are optimized using a master problem [98]. A resource allocation coordination algorithm that is based on the primal decomposition of the overall problem, has already been developed for continuous optimization problems with global capacity constraints by [29].

In fact, a multi-agent control method based on the primal decomposition of the overall control problem will always guarantee the global feasibility of local control decisions. However, since all the control variables are discrete in the control setting considered in the paper, issues such as oscillatory behavior of the discrete decision variables could arise if the resource allocation coordination algorithm is applied directly. As a result, the global optimality of the algorithm cannot be guaranteed anymore.

In this chapter, we develop a mechanism based on the branch-and-bound paradigm [70] to improve the solution found when using the resource allocation coordination algorithm only. More specifically, this is achieved by building the search tree according to the outcome

of the resource allocation coordination algorithm at each node and by returning the best solution found when the overall method stops.

In the literature, a mechanism has been proposed in [19] to deal with the oscillatory behavior of the discrete decision variables. That mechanism is also based on the branch-and-bound paradigm but it uses a distributed algorithm based on the dual decomposition of the overall problem. Since in the dual decomposition approach constraints are relaxed and accounted for in the objective by using penalties for violations, it cannot be guaranteed that the constraints are always satisfied during iterations. Moreover, a general framework of embedded optimization based on the branch-and-bound paradigm has been presented in [49] for model predictive control of hybrid systems, which includes the class of hybrid systems considered in our work. However, that paper focuses on building a problem-specific branch-and-bound search tree by pre-processing heuristics. So, to our best knowledge, an online and problem-independent algorithm for the control of the class of hybrid systems considered here has not yet been proposed in the literature.

In our previous work [77], we have integrated a resource allocation coordination algorithm into the branch-and-bound paradigm. However, in [77], we assumed the solver to have full information of the overall problem, and we did not consider protecting the confidential information of the local subproblems. In the contrast, the main contribution of this chapter consists in reducing information sharing among local control agents, which helps protecting the confidential information of the local subproblems.

With respect to the literature, the multi-agent control method proposed in this chapter only requires limited information sharing among local control agents. In addition, it guarantees the global feasibility of local control decisions and it is also able to efficiently search the overall solution space online by making use of the outcome of the resource allocation coordination algorithm at each node of the tree.

The chapter is organized as follows. In Section 7.2, the considered class of hybrid systems with subsystems governed by discrete inputs and subject to global hard constraints is formalized. In Section 7.3, the resource allocation coordination algorithm based on a projected subgradient method is presented. In Section 7.4, we present the overall proposed multi-agent model predictive control method. In Section 7.5, we consider charging control of a fleet of electric vehicles as an application example of the proposed method and assess the performance of the proposed method in a simulation study. Section 7.6 summarizes the results of this chapter and presents some ideas for future work.

7.2 Model predictive control for a class of hybrid systems

In this section, we focus on the control problem formulation of hybrid systems governed by discrete inputs and subject to global hard constraints. Assume that a large-scale hybrid system consists of N subsystems such that:

- each subsystem is controlled by a control agent
- each control agent has a dynamical model of its subsystem
- each control agent has to solve its local problem
- each agent does not have any information of the models and the local control problems of other subsystems

- subsystems together have to satisfy global hard constraints

7.2.1 Model of subsystem dynamics

Let the dynamics of subsystem i be given by the following deterministic discrete-time model with discrete inputs:

$$x_{i,k+1} = A_i x_{i,k} + B_i u_{i,k} \tag{7.1}$$

$$y_{i,k} = C_i x_{i,k} + D_i u_{i,k} \tag{7.2}$$

where at time step k, for subsystem i, $x_{i,k} \in \mathbb{R}^{n_{i,x}}$ is the local continuous state, $y_{i,k} \in \mathbb{R}^{n_{i,y}}$ the local continuous output, $u_{i,k} \in U_{i,k} \subset \mathbb{R}^{n_{i,u}}$ the local discrete input with $u_{i,k,v} \in U_{i,k,v}$, $U_{i,k,v} \subset \mathbb{R}$ a finite set of scalar values, and $U_{i,k} = U_{i,k,1} \times U_{i,k,2}, ..., U_{i,k,n_{i,u}}$, and $A_i \in \mathbb{R}^{n_{i,x} \times n_{i,x}}$, $B_i \in \mathbb{R}^{n_{i,x} \times n_{i,u}}$, $C_i \in \mathbb{R}^{n_{i,y} \times n_{i,x}}$ and $D_i \in \mathbb{R}^{n_{i,y} \times n_{i,u}}$.

7.2.2 Model predictive control of a single subsystem

In model predictive control, at each control cycle, the control agent determines its local control input by computing the optimal control input sequence over a finite prediction horizon of N_p steps according to an objective function describing the control goals, subject to a model of the subsystem and operational constraints. After that, the control agent applies the first control input in that sequence to its subsystem and waits until the next control cycle starts. For the sake of simplicity of notation, in the following, a bold variable is used to denote the compact expression of variables over the prediction horizon, e.g., $\boldsymbol{x}_{i,k} = [\ x_{i,k}^\mathrm{T} \quad x_{i,k+1}^\mathrm{T} \quad ... \quad x_{i,k+N_\mathrm{p}-1}^\mathrm{T}\ ]^\mathrm{T}$.

Assume that the maximum amount of resources $\theta_{i,k+l} \in \mathbb{R}$ available to subsystem i for $l = 0, ..., N_\mathrm{p} - 1$ is given. Then control agent i makes a measurement of the local state $x_{i,k}$ of its subsystem at time step k. After that, the following optimization problem is solved by control agent i:

$$\begin{aligned} \min_{\boldsymbol{u}_{i,k}} \ & \sum_{l=0}^{N_\mathrm{p}-1} J_i(u_{i,k+l}, x_{i,k+l}) \\ \text{subject to } & x_{i,k+1+l} = A_i x_{i,k+l} + B_i u_{i,k+l} \\ & G_i(u_{i,k+l}) \leq \theta_{i,k+l} \\ & u_{i,k+l} \in U_{i,k+l} \\ & \text{for } l = 0, ..., N_\mathrm{p} - 1 \end{aligned} \tag{7.3}$$

where $J_i : U_{i,k+l} \times \mathbb{R}^{n_{i,x}} \to \mathbb{R}$ is assumed to be a convex function w.r.t. $u_{i,k+l}$ for a given $x_{i,k+l}$ that gives the cost per prediction step and $G_i : U_{i,k+l} \to \mathbb{R}$ is assumed to be a monotonic strictly increasing function w.r.t. $u_{i,k+l}$ that gives the amount of resources consumed by subsystem i per prediction step.

7.2.3 Global constraints

Let r_{k+l} denote the total amount of resources available for all the subsystems at time step $k+l$. Then the global constraints over the prediction horizon are given by

$$\sum_{i=1}^{N} \theta_{i,k+l} = r_{k+l}, \text{ for } l = 0, ..., N_\text{p} - 1 \tag{7.4}$$

7.2.4 Combined overall control problem

We aim to achieve global optimal system performance. Therefore, we define the combined overall control problem by aggregating the local control problems and including the global constraints (7.4), i.e.,

$$\begin{aligned} \min_{\boldsymbol{u}_k, \boldsymbol{\theta}_k} \quad & \sum_{i=1}^{N} \sum_{l=0}^{N_\text{p}-1} J_i(u_{i,k+l}, x_{i,k+l}) \\ \text{subject to} \quad & x_{i,k+1+l} = A_i x_{i,k+l} + B_i u_{i,k+l} \\ & G_i(u_{i,k+l}) \leq \theta_{i,k+l} \\ & u_{i,k+l} \in U_{i,k+l} \\ & \sum_{i=1}^{N} \theta_{i,k+l} = r_{k+l} \\ & \text{for } i = 1, ..., N \text{ and } l = 0, ..., N_\text{p} - 1 \end{aligned} \tag{7.5}$$

where $\boldsymbol{u}_k = [\boldsymbol{u}_{1,k}^\text{T} \ \boldsymbol{u}_{2,k}^\text{T} \ \ldots \ \boldsymbol{u}_{N,k}^\text{T}]^\text{T}$ and $\boldsymbol{\theta}_k = [\boldsymbol{\theta}_{1,k}^\text{T} \ \boldsymbol{\theta}_{2,k}^\text{T} \ \ldots \boldsymbol{\theta}_{N,k}^\text{T}]^\text{T}$.

Since each local agent does not have information of the local models and local problems of other agents, none of the agents has full information of the overall problem. Therefore, it is not possible to solve the combined overall control problem (7.5) in a centralized way. To deal with this issue, in the following sections, we will develop a multi-agent control method to solve the combined overall control problem by introducing a resource allocation coordinator that only requires very limited information from the local agents.

7.3 Resource allocation coordination

Before presenting the overall multi-agent model predictive control method, in the section, we first describe the resource allocation coordination algorithm on which the overall control method will be based. In the resource allocation coordination algorithm, the maximum amount of resources allocated to each subproblem is represented by an auxiliary variable and then the coordination is achieved by solving a master problem that optimizes all the auxiliary variables [98].

7.3.1 Primal decomposition

By dropping $x_{i,k}$ and the time step k and l, problem (7.5) can be simplified as

$$\min_{\boldsymbol{u}, \boldsymbol{\theta}} \quad \sum_{i=1}^{N} J_i(u_i) \tag{7.6}$$

$$\begin{aligned} \text{subject to} \quad & G_i(u_i) \le \theta_i,\ i = 1,...,N \\ & u_i \in U_i,\ i = 1,...,N \\ & \sum_{i=1}^{N} \theta_i = r \end{aligned}$$

Now, let us define

$$p_i(\theta_i) = \min_{u_i \in U_i,\ G_i(u_i) \le \theta_i} J_i(u_i) \tag{7.7}$$

Then, problem (7.6) can be written as

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \sum_{i=1}^{N} p_i(\theta_i) \\ \text{subject to} \quad & \sum_{i=1}^{N} \theta_i = r \end{aligned} \tag{7.8}$$

Problem (7.8) is called the master problem.

7.3.2 Optimization algorithm

Actually, if all optimization variables u_i are continuous, the master problem (7.8) can be solved efficiently by using a subgradient method, which is a simple iterative method for solving convex optimization problems [115]. More specifically, given a convex problem with decision variable u, classical subgradient methods search for the solution to the problem by using the following iteration:

$$u^{(z+1)} = \Pi\Big(u^{(z)} - \alpha^{(z)} h\big(u^{(z)}\big)\Big)$$

where z denotes the iteration step, $h\big(u^{(z)}\big)$ denotes a subgradient of the objective function of the problem at $u^{(z)}$, $\alpha^{(z)}$ denotes the step size at step z, and $\Pi(\cdot)$ denotes the projection onto the constrained solution space.

It can be derived that in problem (7.7), a subgradient of $p_i(\theta_i)$ at θ_i is given by $-\lambda_i$, with λ_i the corresponding Lagrange multiplier to the constraint $G_i(u_i) \le \theta_i$ [15, Chapter 6.4.2]. In particular, the projected subgradient method is given in [29] as

$$\theta_i^{(z+1)} = \theta_i^{(z)} + \xi^{(z)}\Big(\lambda_i^{(z)} - \frac{1}{N}\sum_{j=1}^{N} \lambda_j^{(z)}\Big) \tag{7.9}$$

where $\xi^{(z)}$ is a diminishing step size that satisfies

$$\xi^{(z)} > 0,\ \lim_{z \to +\infty} \xi^{(z)} = 0,\ \sum_{z=1}^{+\infty} \xi^{(z)} = +\infty,\ \sum_{z=1}^{+\infty} (\xi^{(z)})^2 < +\infty \tag{7.10}$$

Note that in (7.9), $-\lambda_i^{(z)}$ is used as the subgradient of $p_i(\cdot)$ at $\theta_i^{(z)}$ and a projection is used to guarantee that the constraint $\sum_{i=1}^{N} \theta_i^{(z)} = r$ is satisfied for all iterations.

In fact, the corresponding λ_i to the constraint $G_i(u_i) \le \theta_i$ in problem (7.7) can in general

Algorithm 7.1 Resource allocation coordination algorithm for problems with discrete optimization variables

Inputs: r, N, ξ and J_i, G_i, U_i for all i

i) Initialize $\theta_i^{(1)}$ for all i and set $z = 1$.

ii) At iteration z, each agent i solves its local problem

$$u_i^{*,(z)} = \underset{u_i \in U_i,\ G_i(u_i) \leq \theta_i^{(z)}}{\arg\min} \quad J_i(u_i)$$

and obtains $\lambda_i^{(z)}$ using (7.11).

iii) Update $\theta_i^{(z+1)}$ for all i using (7.9).

iv) Stop if $|\theta_i^{(z+1)} - \theta_i^{(z)}| \leq \epsilon$ for all i or if the maximum number of iterations is reached; otherwise set $z \leftarrow z+1$ and go back to step ii).

Outputs: $u_i^{*,(z)}$ and $\theta_i^{(z)}$

be computed by

$$\lambda_i = \begin{cases} -\frac{d_{i,\mathbf{J}}^{\mathrm{T}} \cdot d_{i,\mathbf{G}}}{d_{i,\mathbf{G}}^{\mathrm{T}} \cdot d_{i,\mathbf{G}}}, & \text{if} -\frac{d_{i,\mathbf{J}}^{\mathrm{T}} \cdot d_{i,\mathbf{G}}}{d_{i,\mathbf{G}}^{\mathrm{T}} \cdot d_{i,\mathbf{G}}} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{7.11}$$

where $d_{i,\mathbf{J}}$ and $d_{i,\mathbf{G}}$ are respectively the derivatives of $J_i(\cdot)$ and $G_i(\cdot)$ w.r.t. u_i^* with

$$u_i^* = \underset{u_i \in U_i,\ G_i(u_i) \leq \theta_i}{\arg\min} \quad J_i(u_i)$$

More specifically, the explicit resource allocation coordination algorithm for problems with discrete optimization variables is presented in Algorithm 7.1.

7.3.3 Problems arising when applied to optimization problems with discrete decision variables

Actually, if the problem (7.6) is strictly convex, the global optimum is always attained by using the resource allocation coordination algorithm presented above. However, even if $J_i(\cdot)$ and $G_i(\cdot)$ are strictly convex functions, the overall problem is still not convex if u_i is a discrete variable. In fact, in that case, if the resource allocation coordination algorithm is directly applied to the problem, the discrete decision variables may exhibit oscillatory behavior (i.e. the computed optimal values oscillate from one iteration to the next), and hence the global optimum may not be attained. To be more specific, a simple numerical example is given next to show the problem of directly applying the resource allocation coordination algorithm to an optimization problem with discrete decision variables.

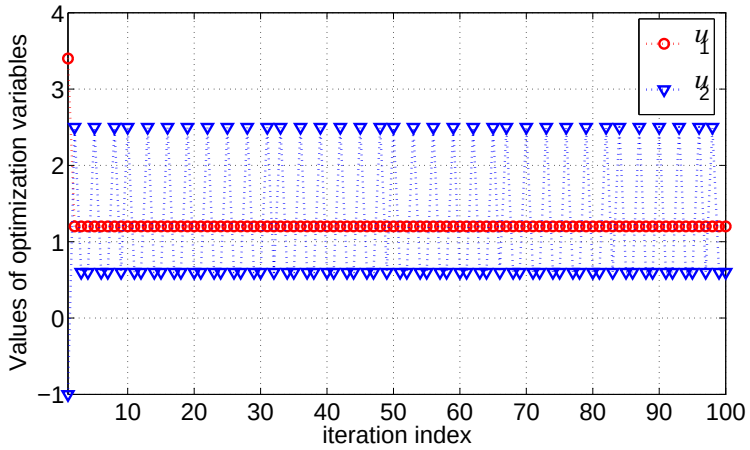

Figure 7.1: Example of oscillatory behavior in the case of discrete optimization variables

Consider the following resource allocation problem:

$$\begin{aligned} \min_{u_1,u_2} \quad & (u_1-3)^2+2(u_2-2)^2 \\ \text{subject to} \quad & u_1 \in \{-1.5, 1.2, 2.4, 3.4, 4.5\} \\ & u_2 \in \{-1, 0.6, 2.5, 3.8, 4.2\} \\ & u_1+u_2 \leq 4.5 \end{aligned}$$

The global optimum to this problem is $u_1^* = 1.2$ and $u_2^* = 2.5$. Figure 7.1 shows the behavior of the values of the discrete decision variables when the resource allocation coordination algorithm is directly applied to this simple example. Note that the dotted line included in the figure is only used to highlight the oscillatory behavior of the value of u_2. It is clearly shown that the value of u_1 converges to 1.2 during the iterations while that of u_2 oscillates between 0.6 and 2.5. Therefore, the global optimum is only part of the oscillation.

7.4 Multi-agent model predictive control method based on resource allocation coordination

Although the values of the control decision variables may oscillate when the resource allocation coordination algorithm is applied to the combined overall control problem (7.5), the oscillations of the values of the control decision variables can be used as a guideline for branching the overall solution space. In this section, we develop a multi-agent control method for the combined overall control problem (7.5) by integrating the resource allocation coordination algorithm into a search tree building mechanism for the overall solution space. The communication structure used in the method is shown in Figure 7.2.

7.4.1 Resource allocation coordinator

The resource allocation coordinator is in charge of allocating the resource to the subsystems. More specifically, receiving $\boldsymbol{\lambda}_{i,k}^{(z)}$ for all i from local agents, the resource

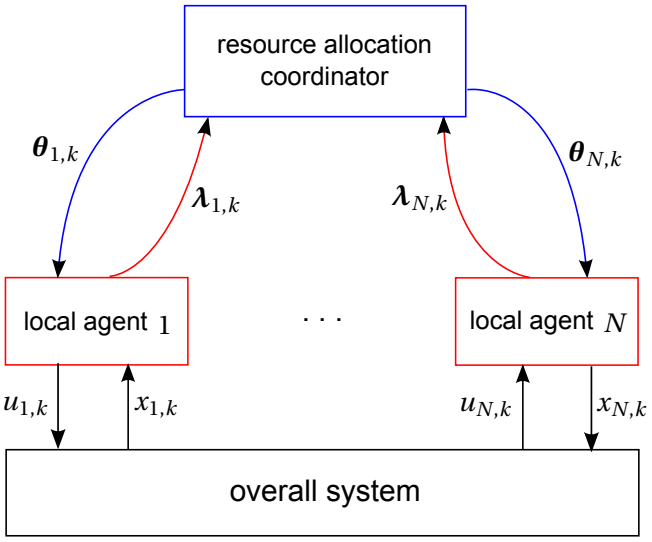

Figure 7.2: Communication structure of the multi-agent model predictive control method

allocation coordinator updates the amounts of resource allocated to local agents by

$$\boldsymbol{\theta}_{i,k}^{(z+1)} = \boldsymbol{\theta}_{i,k}^{(z)} + \xi^{(z)}\left(\boldsymbol{\lambda}_{i,k}^{(z)} - \frac{1}{N}\sum_{j=1}^{N}\boldsymbol{\lambda}_{j,k}^{(z)}\right), \text{ for } i = 1,...,N \tag{7.12}$$

with diminishing step size $\xi^{(z)}$ satisfying (7.10).

7.4.2 Local agent

Each local agent focuses on solving its local control problem and communicating the Lagrange Multiplier associated with the local constraint to the coordinator. More specifically, given the allocated maximal amount $\boldsymbol{\theta}_{i,k}^{(z)}$ of resources from the resource allocation coordinator, each local agent i solves its local control problem (7.3) to obtain $\boldsymbol{u}_{i,k}^{*,(z)}$ and then calculates

$$\lambda_{i,k+l}^{(z)} = \begin{cases} -\frac{d_{i,k+l,\mathbf{J}}^{\mathrm{T}} \cdot d_{i,k+l,\mathbf{G}}}{d_{i,k+l,\mathbf{G}}^{\mathrm{T}} \cdot d_{i,k+l,\mathbf{G}}}, & \text{if } -\frac{d_{i,k+l,\mathbf{J}}^{\mathrm{T}} \cdot d_{i,k+l,\mathbf{G}}}{d_{i,k+l,\mathbf{G}}^{\mathrm{T}} \cdot d_{i,k+l,\mathbf{G}}} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{7.13}$$

where $d_{i,k+l,\mathbf{J}}$ and $d_{i,k+l,\mathbf{G}}$ for $l = 0,...,N_\mathrm{p}-1$ are respectively the derivatives of $J_i(\cdot)$ and $G_i(\cdot)$ w.r.t $u_{i,k+l}$ at $u_{i,k+l}^{*,(z)}$.

7.4.3 Multi-agent control procedure

The overall multi-agent control procedure consists of two levels of subprocedures:

- the lower-level resource allocation coordination subprocedure
- the higher-level subprocedure to branch the solution space and to improve the current best solution to the overall control problem

More specifically, the overall procedure of the multi-agent control method is shown in Figure 7.3.

Lower-level subprocedure

The resource allocation coordination subprocedure (involving the coordinator and the local agents), the schematic representation of which is shown in Figure 7.2, can be explicitly described as follows:

i) The coordinator proposes an initial plan for allocating resources to local agents and communicates the corresponding proposed amount $\boldsymbol{\theta}_{i,k}^{(1)}$ of resources to each local agent i.

ii) At iteration z, each local agent i receives a proposed value $\boldsymbol{\theta}_{i,k}^{(z)}$ from the coordinator and evaluates it by solving its local control problem (7.3). After that, agent i determines $\boldsymbol{\lambda}_{i,k}^{(z)}$ using (7.13) and then communicates it to the coordinator, indicating how much the agent would benefit from extra resources.

iii) At iteration z, based on the proposed value $\boldsymbol{\theta}_{i,k}^{(z)}$ communicated to each local agent i and $\boldsymbol{\lambda}_{i,k}^{(z)}$ received from each local agent i, the coordinator proposes an updated plan of resource allocation $\boldsymbol{\theta}_{i,k}^{(z+1)}$ for all agents using (7.12) and communicates it to the local agents.

iv) During the iterations, the evolution of local control decisions is checked (by the local agents). Depending on whether local control decisions oscillate, two cases can occur:

- In the case where oscillation of local control decisions is detected (see Remark 1), stop the lower-level subprocedure and return:
  - the best proposal of resource allocation (with the lowest sum of the cost functions of local agents) so far
  - the index of the local control decision variable for which oscillation is detected
  - the discrete values between which the given control decision variable oscillates
- In the case where no oscillation of local control decisions is detected:
  - if the allowed maximum number of iterations is reached or $\left|\theta_{i,k+l}^{(z+1)} - \theta_{i,k+l}^{(z)}\right| \leq \epsilon$ holds for all i and for all l, stop the subprocedure and return the best proposal of resource allocation (with the lowest sum of the cost functions of local agents) so far
  - otherwise, set $z \leftarrow z+1$ and go to step ii)

Remark 1.
The oscillation of a discrete decision variable $u_{i,k+l,v}$ is characterized by the following condition, the proof of which can be found in Appendix A:

$$u_{i,k+l,v}^{*,(z+1)} \neq u_{i,k+l,v}^{*,(z)}, \ \operatorname{sgn}\left(\Delta\theta_{i,k+l}^{(z+1)}\right) \neq \operatorname{sgn}\left(\Delta\theta_{i,k+l}^{(z)}\right) \tag{7.14}$$

where $\Delta\theta_{i,k+l}^{(z)} = \theta_{i,k+l}^{(z)} - \theta_{i,k+l}^{(z-1)}$. Therefore, we diagnose the oscillation of discrete decision variables by detecting the condition (7.14) for each i, l and v.

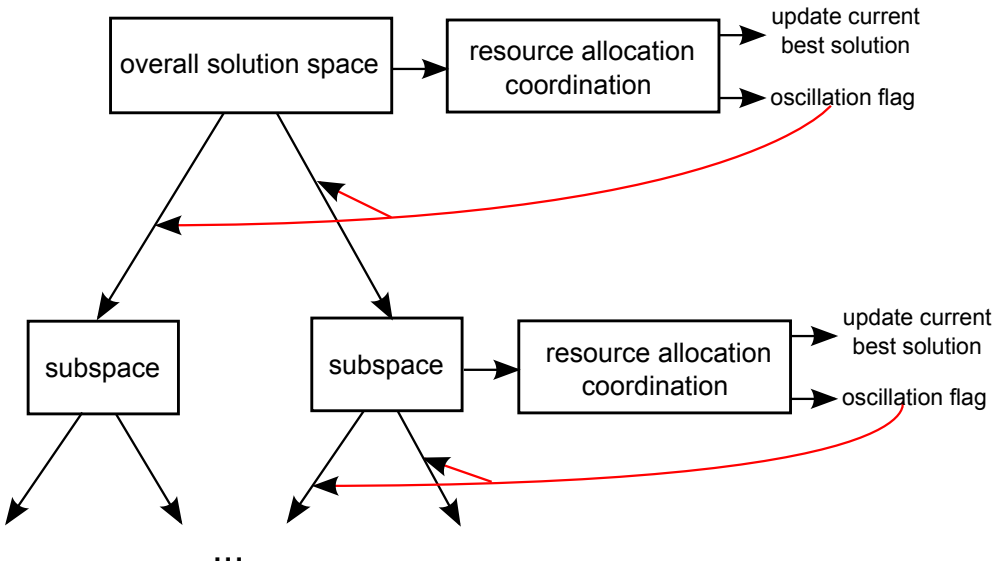

Figure 7.3: Schematic diagram of the higher-level subprocedure

Higher-level subprocedure

The procedure to improve the current best solution, the schematic diagram of which is shown in Figure 7.3, is given by:

a) call the lower-level subprocedure

b) update the best solution to the overall problem found so far by comparing the corresponding values of objective function to the current best solution and the newly found solution by the lower-level subprocedure

c) depending on whether local control decisions oscillate

- if $u_{i,k+l,v}$ is found to oscillate between $\alpha \in U_{i,k+l,v}$ and $\beta \in U_{i,k+l,v}$, separate $U_{i,k+l,v}$ into two sets:
  - set $U^{(1)}_{i,k+l,v} \leftarrow U_{i,k+l,v} \backslash \{\gamma \mid \gamma \in U_{i,k+l,v},\ \gamma \leq \alpha\}$
  - set $U^{(2)}_{i,k+l,v} \leftarrow U_{i,k+l,v} \backslash \{\gamma \mid \gamma \in U_{i,k+l,v},\ \gamma > \alpha\}$

  and solve the two branches in parallel
- if no oscillation of local discrete decision variables is detected, stop and return the best solution found so far

d) when the computation time or number of information exchanges between the coordinator and the local agents reaches a predefined upper bound or $U_{i,k+l,v}$ for all i, l, v in all calls of the lower-level sub-procedure has only one single element, stop the procedure.

7.5 Charging control of electric vehicles

As an example of the developed multi-agent model predictive control method, in this section we address the charging control of a fleet of electric vehicles under constrained grid conditions.

Due to their higher energy efficiency and lower emission of pollutants, electric vehicles are used more and more. Charging this increasing number of electric vehicles will inevitably cause additional load to the electrical power distribution grid [58]. Therefore, a smart charging control strategy that balances the charging demands of electric vehicles is highly preferred by the distribution grid operators. So far, intelligent charging control of electric vehicles has been addressed by using distributed integer linear optimization [130], sequential quadratic programming [28, 52], dynamic programming [53], and heuristic methods [111]. In this section, we assume that each electric vehicle is equipped with a charging controller and each controller only shares limited information with the external environment, and apply the proposed multi-agent model predictive control method to the charging of a fleet of electric vehicles under constrained grid conditions.

We focus on the optimal charging control of a fleet of electric vehicles at a charging station within a given time period, e.g. a day. We assume that there is a charging point for each vehicle in the station. Given the profile of the electricity price, the arrival and the departure times of all electric vehicles at the charging station, and the maximum power limit provided by the grid, we aim to charge all electric vehicles up to the required levels while minimizing the total cost of electricity use.

7.5.1 Definitions

We define k as the discrete-time step counter, T as the length of the simulation time interval, with a typical value of 15 minutes, and k_{d} as the last step of the overall charging period. Then define N_{v} as the total number of electric vehicles under consideration. Define $T_{i,\mathrm{arrival}}$, $T_{i,\mathrm{departure}}$ as the arrival time and the departure time of electric vehicle i at the charging station, respectively. Without loss of generality, we assume $T_{i,\mathrm{arrival}}$ and $T_{i,\mathrm{departure}}$ are integer multiples[1] of T. Define $s_{i,k}$ as the state of charge of electric vehicle i at time kT and s_i^{req} as the required state of charge of electric vehicle i before departing from the charging station. Define C_i^{p} as the capacity of the battery of electric vehicle i and $d_{i,\mathrm{tol}}$ as the allowed tolerance on the difference between the state of charge of electric vehicle i at its departure time and s_i^{req}. Besides, we assume the charging power of each electric vehicle within a simulation interval is constant and define $p_{i,k}$ as the amount of power consumed by electric vehicle i in the time interval $\big[kT,(k+1)T\big)$. Finally, we define $u_{i,k}$ as the binary control variable indicating whether electric vehicle i is charging in the time interval $\big[kT,(k+1)T\big)$.

7.5.2 Model of the charging of an individual electric vehicle

First, the amount of power consumed by an electric vehicle i within the time interval $[kT,(k+1)T]$ is given by

$$p_{i,k}=\begin{cases} F\big(s_{i,k}\big), & \text{if } u_{i,k}=1 \\ 0, & \text{if } u_{i,k}=0 \end{cases} \tag{7.15}$$

[1] If an electric vehicle arrives earlier or departs later than a sampling time instant, it will not be charging in the partial time slot within which it arrives or departs.

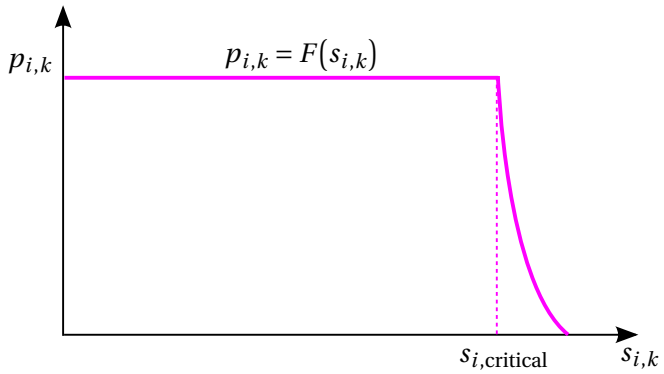

Figure 7.4: Electric vehicle charging using the CPCV option. With this option, the vehicle is first charging with constant power until the critical state of charge s_{critical} is reached. After that, it is charged with constant voltage until its battery is fully charged.

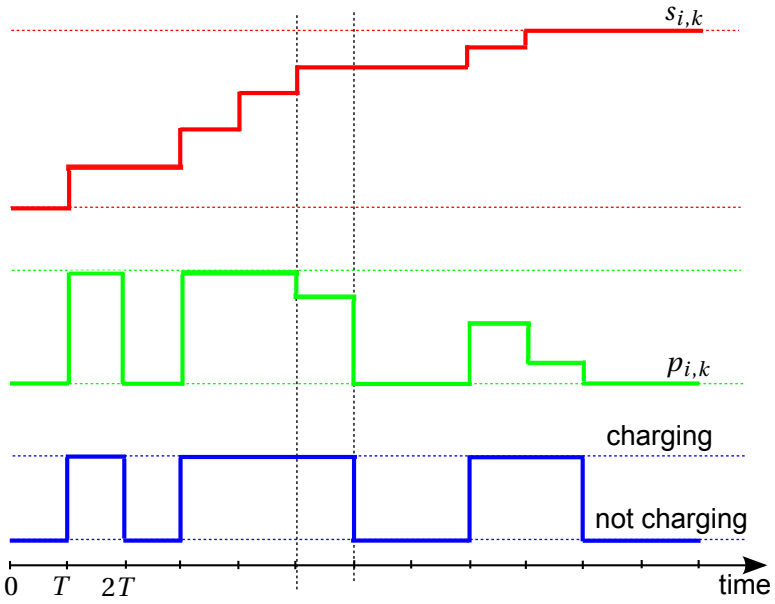

Figure 7.5: Charging profile of an electric vehicle under charging control

where the function $F(\cdot)$ describes how the amount of power consumed by electric vehicle i depends on its state of charge. Next, the state of charge of electric vehicle i is updated by

$$p_{i,k} = F(s_{i,k}) \cdot u_{i,k} \tag{7.16}$$

$$s_{i,k+1} = s_{i,k} + \frac{p_{i,k} \cdot T}{C_i^{\text{p}}} \tag{7.17}$$

There are in general two charging options available for electric vehicle chargers [90], namely Constant Current-Constant Voltage (CCCV) and Constant Power-Constant Voltage (CPCV). In our work, we assume all electric vehicles are charged using the CPCV option. More specifically, the profile of the function $F(\cdot)$ for the CPCV option is shown in Figure 7.4. Moreover, in order to make the model of the charging of an electric vehicle clearer, Figure 7.5 shows the dynamics of the state of charge of an electric vehicle under charging control.

7.5.3 Global constraints

At any time, the total amount of power consumed by all electric vehicles must not exceed the maximum amount of power that can be provided by the grid. Therefore, the constraints imposed by the capacity of the grid on all electric vehicles are given by

$$\sum_{i=1}^{N_\mathrm{v}} p_{i,k} \leq P_{k,\max}, \quad k = 1, ..., k_\mathrm{d} \tag{7.18}$$

where $P_{k,\max}$ denotes the maximum power limit provided by the grid, which can be steady or time-variant.

7.5.4 Charging cost

If the profile of the price of electricity is given, the total cost on charging all electric vehicles is given by:

$$J = \sum_{i=1}^{N_\mathrm{v}} \sum_{k=k_{i,\mathrm{arrival}}}^{k_{i,\mathrm{departure}}-1} p_{i,k} \cdot T \cdot c_k \tag{7.19}$$

where c_k denotes the price of electricity at in the time interval $\big[kT, (k+1)T\big)$ and

$$k_{i,\mathrm{arrival}} = \frac{T_{i,\mathrm{arrival}}}{T}, \quad k_{i,\mathrm{departure}} = \frac{T_{i,\mathrm{departure}}}{T}$$

7.5.5 Problem formulation

Normally, the state of charge of a battery is limited to the interval $[0.2, 0.9]$ [90]. This mainly relates to battery life time aspects: charging the remaining 10%-20% before fully charged has shown to result in quicker battery degradation [90]. According to [90], if the CPCV option is used, an electric vehicle can be charged with constant power up to more than 90% of the capacity of its onboard battery. As we adopt the CPCV option, if we define $s_{i,\mathrm{critical}}$ as the critical state of charge of electric vehicle i, according to [90], we have $s_{i,\mathrm{critical}} > 0.9$ for all i. In order to make the life time of a battery longer, the state of charge of the battery should be limited at most to 0.9. Therefore, we assume that the user always selects s_i^{req} such that $s_i^{\mathrm{req}} \leq 0.9$. Hence, assuming s_i^{req} is sufficiently smaller than $s_{i,\mathrm{critical}}$ and T is not too large, we can assume that

$$s_{i,k} \leq s_{i,\mathrm{critical}} \tag{7.20}$$

holds for all k and for all i. Then, the function $F(\cdot)$ with the CPCV option can be simplified as

$$F\big(s_{i,k}\big) = \begin{cases} p_{i,\mathrm{constant}}, & \text{if } s_{i,k} \leq s_{i,\mathrm{critical}} \\ 0, & \text{otherwise} \end{cases}$$

The charging model (7.16)-(7.17) of an electric vehicle can then be simplified to

$$s_{i,k_{i,\mathrm{departure}}} = s_{i,k_{i,\mathrm{arrival}}} + \sum_{k=k_{i,\mathrm{arrival}}}^{k_{i,\mathrm{departure}}-1} \frac{p_{i,\mathrm{constant}} \cdot u_{i,k} \cdot T}{C_i^{\mathrm{p}}}$$

and the constraint $s_{i,k_{i,\text{departure}}} \geq s_i^{\text{req}} - d_{i,\text{tol}}$, which presents the charging requirement for each electric vehicle i, can be written as

$$\frac{s_i^{\text{req}} - s_{i,k_{i,\text{arrival}}} - d_{i,\text{tol}}}{p_{i,\text{constant}} \cdot T} \cdot C_i^{\text{p}} \leq \sum_{k=k_{i,\text{arrival}}}^{k_{i,\text{departure}}-1} u_{i,k} \tag{7.21}$$

Further, since all the control variables are all binary values, the constraint (7.21) can be rewritten as

$$\sum_{k=k_{i,\text{arrival}}}^{k_{i,\text{departure}}-1} u_{i,k} \geq m_i \tag{7.22}$$

where m_i is an integer constant given by

$$m_i = \text{ceiling}\left(\frac{s_i^{\text{req}} - s_{i,k_{i,\text{arrival}}} - d_{i,\text{tol}}}{p_{i,\text{constant}} \cdot T} \cdot C_i^{\text{p}}\right)$$

Finally, including for all i the constraint (7.22) via penalty term to the objective function with a sufficiently large weight β_i and a normalization factor $1/(k_{i,\text{departure}} - k_{i,\text{arrival}})$, the optimal charging control problem of a fleet of electric vehicles under constrained grid conditions can be formulated as

$$\begin{aligned} \min_{\boldsymbol{u}} \ & \sum_{i=1}^{N_\text{v}} \Bigg(\frac{1}{J_{i,\text{cost,typical}}} \sum_{k=k_{i,\text{arrival}}}^{k_{i,\text{departure}}-1} p_{i,\text{constant}} \cdot T \cdot c_k \cdot u_{i,k} \\ & \qquad + \frac{\beta_i}{k_{i,\text{departure}} - k_{i,\text{arrival}}} \Big| m_i - \sum_{k=k_{i,\text{arrival}}}^{k_{i,\text{departure}}-1} u_{i,k} \Big| \Bigg) \\ \text{subject to } & \sum_{i=1}^{N_\text{v}} p_{i,\text{constant}} \cdot u_{i,k} \leq P_{k,\max}, \ k = 1, \ldots, k_\text{d} \end{aligned} \tag{7.23}$$

where $\boldsymbol{u} = [\boldsymbol{u}_1^\text{T} \ \boldsymbol{u}_2^\text{T} \ \ldots \ \boldsymbol{u}_{N_\text{v}}^\text{T}]^\text{T}$ with $\boldsymbol{u}_i = [u_{i,k_{i,\text{arrival}}} \ \ldots \ u_{i,k_{i,\text{departure}}-1}]^\text{T}$, and $J_{i,\text{cost,tyical}}$ is the typical value for the charging cost of electric vehicle i, which is used to normalize the real charging cost of electric vehicle i. Alternatively, $J_{i,\text{cost,tyical}}$ can be given by cost related to the average or maximum number of steps needed to charge from the current state of charge to the required level. Note that problem (7.23) is a specific case of the general combined control problem (7.5). Therefore, the proposed multi-agent control method is applicable to the optimal charging control problem of electric vehicles.

7.5.6 Numerical simulation study

Now we apply the proposed multi-agent control method to the charging control problem (7.23) of electric vehicles. In this simulation study, we consider two cases where respectively 5 and 20 electric vehicles need to be charged. With the simpler Case 1, we aim to show that the proposed multi-agent control method finds the global optimal solution while with the more complicated Case 2, we aim to show the flexibility and effectiveness of the proposed method when limiting the computation and communication budget. The information of all vehicles in both cases is summarized in Tables 7.1 and 7.2, respectively.

Table 7.1: Data for 5 electric vehicles charging with $P^{\max} = 8\text{kW}$ in Case 1.

i	k_{arrival}	$k_{\text{departure}}$	s_{initial}	s^{req}	C^{p} (kWh)	p_{constant} (kW)
1	3	6	0.60	0.80	9	3.5
2	1	4	0.35	0.45	7.1	2.5
3	2	5	0.40	0.60	8	3
4	5	10	0.60	0.90	8.5	2.7
5	4	8	0.50	0.70	7.5	3.2

Table 7.2: Data for 20 electric vehicles charging with $P^{\max} = 36\text{kW}$ in Case 2.

i	k_{arrival}	$k_{\text{departure}}$	s_{initial}	s^{req}	C^{p} (kWh)	p_{constant} (kW)
1	3	6	0.60	0.80	9	3.5
2	1	4	0.35	0.45	7.1	2.5
3	2	5	0.40	0.60	8	3
4	5	10	0.60	0.90	8.5	2.7
5	4	8	0.50	0.70	7.5	3.2
6	3	9	0.30	0.50	7.8	3.5
7	2	7	0.45	0.75	8.3	2.9
8	1	5	0.60	0.80	8.6	3.1
9	2	6	0.40	0.65	7.7	3.4
10	6	10	0.50	0.75	8.8	3.7
11	4	8	0.65	0.85	8.6	3.2
12	2	6	0.43	0.63	7.5	2.4
13	1	5	0.38	0.58	8.2	3.1
14	2	7	0.55	0.85	8.8	2.8
15	2	6	0.40	0.60	7.6	3.1
16	1	9	0.45	0.65	7.9	3.3
17	3	9	0.50	0.80	8.4	2.8
18	3	8	0.60	0.85	8.5	3
19	2	6	0.55	0.70	7.6	3.1
20	7	11	0.60	0.85	8.7	3.3

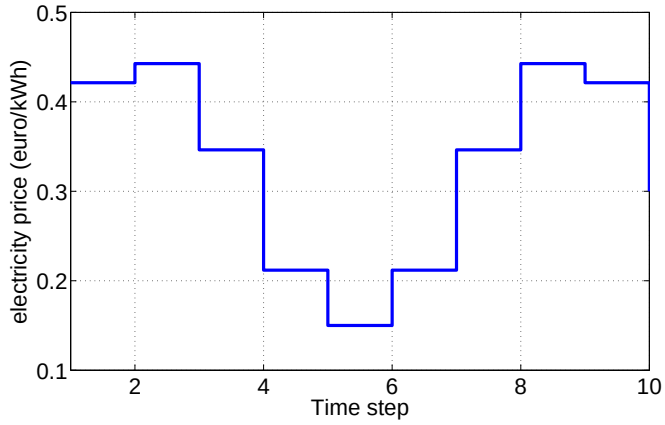

Figure 7.6: Profile of the electricity price for Case 1

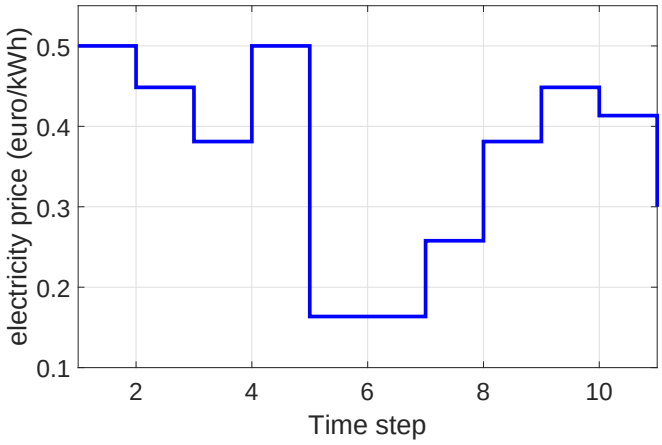

Figure 7.7: Profile of the electricity price for Case 2

The total number of binary control variables in Case 1 is $\sum_{i=1}^{5}(k_{i,\text{departure}} - k_{i,\text{arrival}}) = 18$ while in Case 2 it is $\sum_{i=1}^{20}(k_{i,\text{departure}} - k_{i,\text{arrival}}) = 89$. The parameters used in the simulations are $T = 15$ min, $d_{i,\text{tol}} = 0.02$, and $\beta_i = 200$ for all i. Besides, for each electric vehicle i, $J_{i,\text{cost,typical}} = \left(k_{i,\text{departure}} - k_{i,\text{arrival}}\right) \cdot T \cdot \bar{c} \cdot p_{i,\text{constant}}$ with $\bar{c}$ denoting the average price of electricity for the simulated period; the maximum number of iterations in the resource allocation coordination algorithm is set to 1000 and $\epsilon = 0.001$; the simulated period is 165 minutes and the profiles of the electricity price for Case 1 and Case 2 are shown in Figures 7.6 and 7.7. The simulations are performed using Matlab 2013b on a cluster computer consisting of 4 blades with 2 eight-core 3.3 GHz E5-2643 processors, and 64 GiB memory per blade.

In the simulation for Case 1, no limit is imposed on the maximum computation time or on the maximum number of information exchanges between the coordinator and the local control agents. The simulation results are summarized in Table 7.3. Note that the overall problem is a mixed integer linear programming problem and therefore it can be solved efficiently in a centralized way by using state-of-the-art solvers like CPLEX, GUROBI, MOSEK and XPRESS, which yield the globally optimal solution, provided the overall problem would be fully known by a single control agent. However, in the control setting we consider in this chapter, local control agents only share limited information with the

Table 7.3: Simulation results for Case 1

	CPLEX	proposed multi-agent control method
J_{opt}	2.3907	2.3907

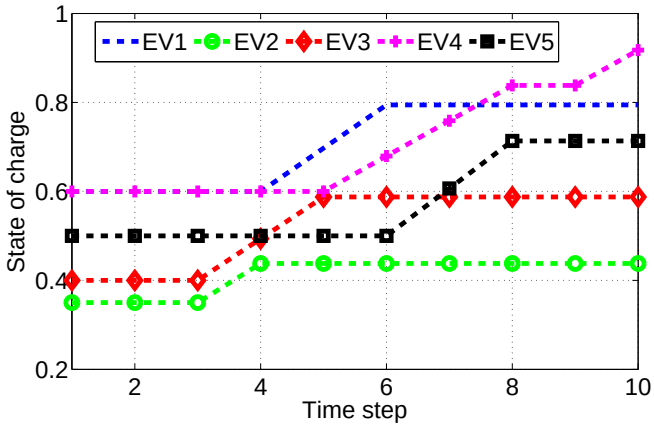

Figure 7.8: Charging dynamics of the 5 electric vehicles with the proposed multi-agent charging control method in Case 1

coordinator. Therefore, neither the coordinator nor any of the local control agents have full information of the overall problem. In Table 7.3, the solution found by using CPLEX for this case is included to validate the solution found by the proposed multi-agent control method. In fact, from Table 7.3, it is clearly seen that the proposed multi-agent control method finds the globally optimal solution for Case 1. Besides, the computation time of the proposed method is 68.76 seconds. Finally, the charging dynamics and the total power consumption of the 5 electric vehicles with the proposed multi-agent charging control in this case are shown in Figures 7.8 and 7.9, respectively. It is clearly seen that the electric vehicles are charged up to the required level without exceeding the maximum power limit provided by the grid, and that they are charged as much as possible when the price of electricity is low.

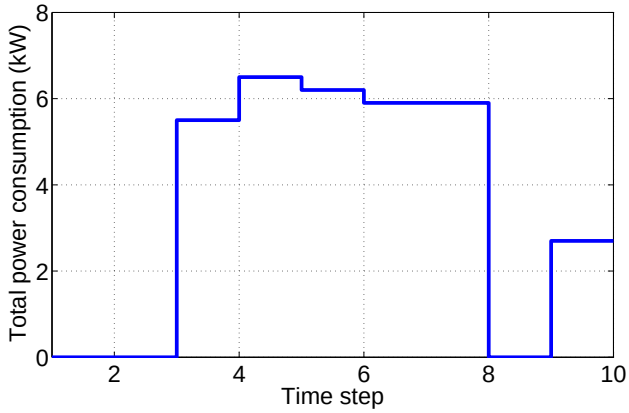

Figure 7.9: Total power consumption of the 5 electric vehicles with the proposed multi-agent charging control method in Case 1

Table 7.4: Simulation results for Case 2

	CPLEX	A1	A2	A3	A4
J_{opt}	9.1946	75.9733	9.6983	75.9733	9.6983

In the simulations for Case 2, the proposed method without limiting the computation time or the number of information exchanges between the coordinator and the local control agents takes too much computation time. Therefore, we decided to use four alternatives that generate feasible solutions in a limited time:

- Alternative 1 (A1): depth-first search; $M_{\max} = 300000$; $t_{\max} = \infty$
- Alternative 2 (A2): breadth-first search; $M_{\max} = 300000$; $t_{\max} = \infty$
- Alternative 3 (A3): depth-first search; $t_{\max} = 60000$ s; $M_{\max} = \infty$
- Alternative 4 (A4): breadth-first search; $t_{\max} = 60000$ s; $M_{\max} = \infty$

where $t_{\max}$ denotes the maximum computation time (in seconds) and $M_{\max}$ denotes the maximum number of information exchanges between the coordinator and the local control agents. Note that in the depth-first search approach, the search tree is explored as far as possible along each branch before backtracking, while in breadth-first search, the search tree is explored as widely as possible on each level of nodes before moving to the next level.

The simulation results of using all the four variants of the proposed method are summarized in Table 7.4. Note that the result found by using CPLEX for this case is also included in Table 7.4 to help indicate how far away the solutions found by the four alternatives are from the globally optimal one. Besides, Figure 7.10 shows the evolution of the values of the overall objective function as a function of the number of nodes that have been visited in the search tree for each of the alternatives. It can be seen from Table 7.4 and Figure 7.10 that although the solutions found by the variants of the proposed multi-agent control method are not the same as the globally optimal solution, the best values of the overall objective function found by the second and the fourth alternatives are close to the globally optimal one. Moreover, given the same computation and communication budget, in the proposed multi-agent control method, breadth-first search generates better results than depth-first search.

Next, in order to show the effectiveness of the proposed multi-agent control method in balancing the quality of solution and the computation time, we conducted another simulation for Case 2 using breadth-first search, $M_{\max} = \infty$ and a lower value for $t_{\max}$ of $t_{\max} = 1200$ s (i.e. 20 minutes). The evolution of the values of the overall objective function in this simulation is shown in Figure 7.11. More precisely, the best value of the overall objective function found using breadth-first search in the given 20 minutes of computation time is 9.7270 and the corresponding charging dynamics and total power consumption of the 20 electric vehicles are shown in Figures 7.12 and 7.13, respectively. Note that in order to assess how far the behavior of the proposed controller is from optimality, we have compared the total power consumption of the electric vehicles with the proposed control method and with centralized optimal control using CPLEX in Figure 7.13. It is seen that the proposed multi-agent controllers show a behavior that is similar to the one of the

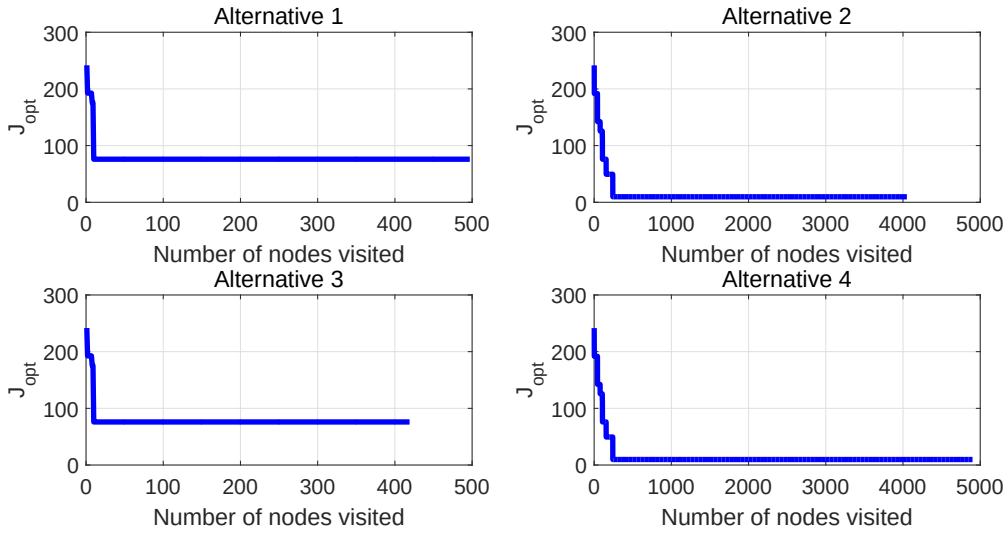

Figure 7.10: Evolution of J_{opt} as a function of the number of nodes visited in the search tree for the alternatives for Case 2

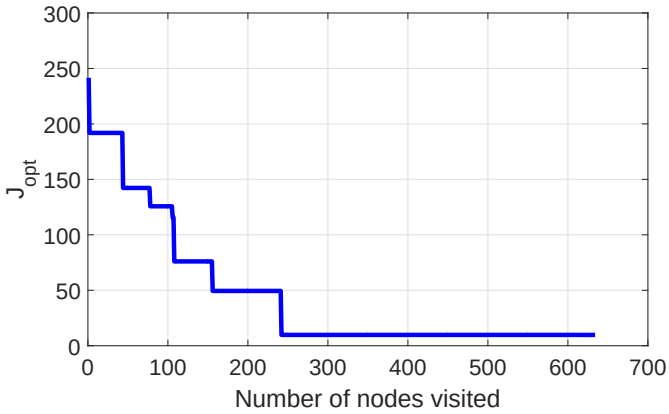

Figure 7.11: Evolution of J_{opt} as a function of the number of nodes visited in the search tree for Case 2 using breadth-first search with $t_{\max} = 1200s$ and $M_{\max} = \infty$.

centralized optimal controller with only slight differences. Besides, the best value of the overall objective function found using breadth-first search in 20 minutes of computation time is close to the globally optimal one and the electric vehicles are all charged up to the required levels without exceeding the maximum power limit imposed by the grid. Therefore, although each control agent only communicates very limited information (i.e. only the Lagrange multiplier associated with the charging power constraint) to the coordinator and only limited computation time is available, the proposed multi-agent control method can still effectively balance the solution quality and the computation time.

Finally, as a first step to investigate the robustness of the proposed multi-agent control method, we consider that the power supply provided by the grid is disturbed and perform a closed-loop receding horizon control simulation for Case 2. More specifically, we consider the power supply predicted by the controllers is constantly $P_{k,\max}^{\text{predicted}} = 36$ kW while the actual

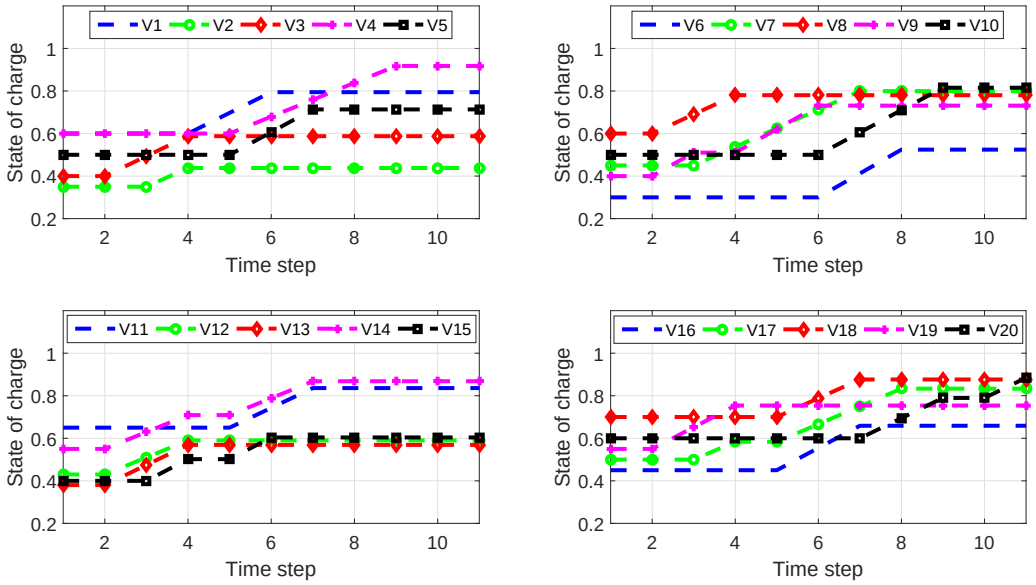

Figure 7.12: Charging dynamics of the 20 electric vehicles with the proposed multi-agent charging control method in Case 2 given 20 minutes of computation time

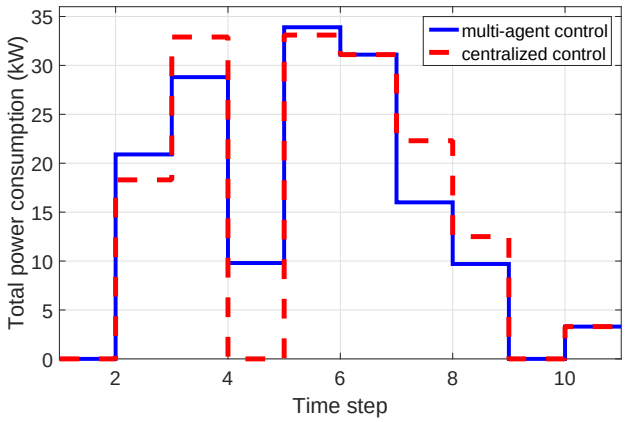

Figure 7.13: Total power consumption of the 20 electric vehicles with the proposed multi-agent charging control method given 20 minutes of computation time and with centralized optimal control for Case 2

power supply at each step k is given by

$$P_{k,\max}^{\text{actual}} = 36 + \omega \quad [\text{kW}]$$

where ω is a normally distributed pseudorandom number with mean 0 and standard deviation 5. The resulting power consumption of all electric vehicles with the proposed multi-agent charging control method, the predicted power supply, and the actual power supply are plotted in Figure 7.14. In order to highlight the effect of the disturbance on the actual power consumption of the vehicles, we also plot the power consumption of vehicles (indicated by red dashed line) in the undisturbed case. It is seen that when the actual power supply is higher than the level that is needed for the control action to be implemented, the

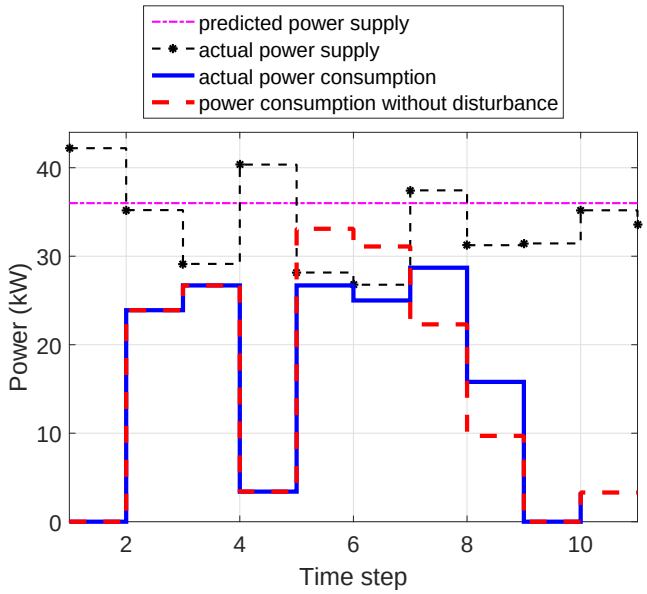

Figure 7.14: Power consumption of the 20 electric vehicles with the proposed multi-agent control method given 20 minutes of computation time in Case 2 when the power supply is disturbed

vehicles are charged with the control action of the proposed method fully implemented. When the actual power supply is lower than the needed level, the control action of the proposed method is not fully implemented in order to respect the actual charging constraints. In that case, the control action of the proposed method is implemented in a way that the vehicles with higher charging-emergency-rate have higher priority to change. Note that in this simulation, the charging-emergency-rate $\varrho_{i,k}$ of a vehicle i at a simulation step k is defined by

$$\varrho_{i,k} = \frac{\text{remaining state of charge}}{\text{remaining charging time}} = \frac{s_i^{\text{req}} - s_{i,k}}{k_{i,\text{departure}} - k}$$

In this way, the actual power consumption is always lower than the actual power supply provided by the grid. Actually, the vehicles that are not charged due to insufficient power supply at certain time steps are charged at later time steps under the control of the proposed method. Besides, the value of the cost function for the disturbed case is 9.6948, which corresponds to a small performance drop of about 3.3% compared with the value 9.3865 of the cost function for the undisturbed case. Therefore, the proposed multi-agent control method is still working adequately when the power supply is disturbed.

7.6 Summary

We have considered multi-agent model predictive control for a class of hybrid systems governed by discrete inputs and subject to global hard constraints where each subsystem has a local convex objective function and a strictly increasing constraint function. We focused on the scenario where each subsystem only shares limited information with the external environment, and we developed a novel multi-agent model predictive control

method by integrating a distributed resource allocation coordination algorithm into a solution space branching mechanism. With the distributed resource allocation algorithm, the global feasibility of the local control decisions is always guaranteed. With the solution space branching mechanism, the search tree for the overall solution space is built smartly based on the outcome of the distributed resource allocation coordination algorithm. Results for the charging control of a fleet of electric vehicles in a simulation study show that the proposed multi-agent control method effectively balances the solution quality, the computation time, and the communication burden.

Possible extensions of the work presented in this chapter include developing a distributed algorithm to compute the global lower bound of the overall objective function, and combining that algorithm with the search tree building mechanism developed in this chapter. In addition, more complex and more challenging case studies considering coordinating both charging-to-vehicle (G2V) and discharging-to-grid (V2G) of electric vehicles may be performed. An extensive assessment including the stability and the robustness of the developed multi-agent control method can also be performed.

Chapter 8

Conclusions and Future Research

This thesis has presented efficient control approaches for urban transportation networks and for hybrid systems with limited information sharing. In this chapter, we highlight the contributions of the thesis and summarize some open issues that can be investigated in the future.

8.1 Contributions of the thesis

The contributions of the thesis can be summarized as follows:

- **Models of the dynamics and the energy consumption of cybercars have been developed.**

  To facilitate the development of efficient control methods for cybercars, in Chapter 3, we have developed a discrete-time model and a discrete-event model for the dynamics and the energy consumption of cybercars.

  The discrete-time model is straightforward to derive. In contrast, the discrete-event model involves checking the order of occurrence of events. The benefit of using the discrete-event model lies in the fact that it is more efficient than the discrete-time model when the number of cybercars is small. However, we focus on a cybernetic transportation network with a large number of cybercars. Therefore, it is more efficient to use the discrete-time modeling method to describe the dynamics and the energy consumption of the cybercars.

- **Efficient and scalable multi-agent dynamic routing methods for cybercars have been developed.**

  Although control of a large number of cybercars can be straightforwardly addressed in a centralized control setting, for reasons of scalability and fast computation, a centralized control method will not be tractable for the large-scale use of cybercars in the future. To contribute to the development of cybernetic transportation systems, in Chapter 4, we have focused on a specific case of the fleet control problem, e.i. dynamic routing, of cybercars, and we have developed several tractable and scalable multi-agent control methods, including multi-agent model predictive control and parameterized control, for solving the problem.

  We have implemented simulation case studies for comparing the performance of the all the developed control methods. The simulation results have indicated that the

flow-splitting-rate-based parameterized control method performs the best among all the proposed multi-agent control methods. Remarkably, for the given case study the flow-splitting-rate-based parameterized control method shows comparable control performance to that of centralized model predictive control while requiring much less online computation time. Besides, the flow-splitting-rate-based parameterized control method can be easily applied to road networks with arbitrary topology. Therefore, we have concluded that the flow splitting rate based parameterized control method is effective in solving the dynamic routing problem of a fleet of cybercars.

- **Efficient methods for routing of traffic flows in urban transportation networks have been developed.**

  To reduce the computational efforts in solving the dynamic routing problem of traffic flows in large-scale urban transportation networks, in Chapter 5, we have proposed a novel hierarchical control approach based on network division and a novel bi-level control approach based on merging nodes and links.

  We have also implemented a simulation-based case study for comparing the hierarchical control approach and the bi-level control approach. It has been shown by the case study that the hierarchical control approach is more effective than the bi-level control approach in solving the routing problem of traffic flows. Besides, the hierarchical control approach is more flexible and can be extended to include multiple control layers.

- **A novel formulation for co-optimization of the orientation of road sections and the routes of traffic flows has been developed.**

  In Chapter 6, we have assumed that the orientation of each road section in an urban road network can be changed in each control period and we have addressed the co-optimization problem that jointly determines the orientation of road sections in the network and routes of traffic flows. Although the co-optimization problem for a large-scale network contains a large number of binary optimization variables, which makes the problem computationally hard to solve, we have developed a novel formulation for the co-optimization problem by considering a circular orientation in each elementary cycle of the network and mapping orientations of road sections using the orientation of these elementary cycles.

  With the formulation developed in Chapter 6, the number of binary variables involved in the co-optimization problem is reduced substantially w.r.t. considering the orientation of each road section independently. It has been shown by a simulation-based case study that the proposed formulation is suitable for on-line optimization.

- **Multi-agent model predictive control for a class of hybrid systems with limited information sharing has been developed.**

  In Chapter 7, we have developed a multi-agent model predictive control method for a class of hybrid systems based on the integration of a distributed resource allocation coordination algorithm and a solution space branching mechanism. We have focused on the scenario where each subsystem only shares limited information with the external environment. The proposed multi-agent control method has two main

advantages. First, thanks to the distributed resource allocation algorithm, which is based on the primal decomposition of the global constraint, the global feasibility of the local control decisions is always guaranteed. Second, with the solution space branching mechanism, the search tree for the overall solution space is built based on the outcome of the distributed resource allocation coordination algorithm.

We have applied the proposed multi-agent model predictive control method to the charging control of a fleet of electric vehicles. The simulation results have showed that the proposed multi-agent control method effectively balances the solution quality, the computation time, and the communication burden.

The work of this thesis has been presented in three journal papers [79–81] and three international conference papers [76, 77, 82].

8.2 Recommendations for future work

In this section, we recommend some possible topics for future research based on the work presented in this thesis.

- **More efficient modeling of the dynamics and the energy consumption of a fleet of cybercars**

  In Chapter 3, we have developed a discrete-time model and a discrete-event model for the dynamics and the energy consumption of cybercars. Due to their difference in determining the time instants for updating the states of the system, the discrete-time model is more efficient when the number of cybercars is large while the discrete-event model is more efficient when the number of cybercars is small.

  Since we focus on the dynamic routing of a large fleet of cybercars, we have used the discrete-time model in the design of the routing controllers. However, it has been observed that the frequency of occurrence of events does not have to be high even if the number of cybercars is large. For instance, if the positions of all the cybercars in the network are far from the ends of the segments that they are running in, even if the number of cybercars is large, the time length between the current moment and the occurrence time of the next event is still longer than the length of a simulation time interval in the discrete-time modeling approach. In that case, it is not efficient to only use the discrete-time modeling method. One possible way to address this issue is to make the most out of the discrete-time model and the discrete-event model by combining both of them. More specifically, we can propose a condition to determine which modeling method is potentially more time-efficient in predicting the future dynamics of the system. After that, at each control step, the more time-efficient model is selected by checking the condition.

- **Combining the dynamic routing of cybercars with the scheduling of cybercars**

  In Chapter 4, we have addressed the dynamic routing of a fleet of cybercars by assuming that there are higher-level controllers assigning transport service requests, including starting time, origin and destination, to each cybercar. We have also assumed that there is a sufficient number of cybercars and that each cybercar leaves the network after delivering a transport service.

To make our work more of practical value, we need to consider that there is a given fixed number of cybercars in the network. Thus, we need to consider combining the dynamic routing of cybercars with the scheduling of them. More specifically, at each control step, for each cybercar still delivering transport service or on the way to pick up next passengers we need to update their routes while for each cybercar that has finished delivering a transport service we need to determine which transport service the cybercar should deliver next or whether the cybercar should be charged considering the energy level of its onboard battery. Besides, we need to consider minimizing a combined system cost consisting of the total time spent by all passengers (including waiting time for picking up and transport service delivery time) and the total energy consumption of all the cybercars (including delivering service and charging costs, etc). Further, we need to develop efficient solution methods, e.g. parameterized control approaches, for the combined problem.

- **Distributed dynamic routing of cybercars based on alternative distributed model predictive control algorithms**

  In Chapter 4, we have applied the distributed model predictive control algorithm presented in [95] to the dynamic routing of cybercars. However, it has been shown by the simulation results that the dynamic routing method based on that distributed model predictive control algorithm does not show comparable control performance with respect to that of the centralized model predictive control method.

  In order to fully explore the potential of distributed model predictive control on the dynamic routing of cybercars, it is recommended that other distributed model predictive control algorithms [85], such as DMPC based on alternative direction multiplier method (ADMM) [23], feasible cooperation-MPC (FC-MPC) [119], and distributed MPC based on a cooperative game [86], etc, should also be adapted to the distributed dynamic routing of cybercars and be evaluated extensively by means of simulation case studies.

- **Further assessment of the routing methods for traffic flows developed in this thesis**

  For the hierarchical control approach and the bi-level control approach developed in Chapter 5, we have performed a simulation case study to compare their control performance and computational speed. It has been suggested by the simulation results that the hierarchical control approach is more efficient than the bi-level control approach in the sense of yielding a better control performance and requiring less computation time. Actually, the advantage of the bi-level control approach over the hierarchical control approach is that it requires less communication efforts for transferring the states of the network to the control agents. However, in the case study, which serves as our first step to evaluate the two approaches, we have not yet considered the comparison of the required communication efforts of the two approaches.

  In order to assess the performance of the two approaches comprehensively, it is recommended that more detailed case studies, where comparisons of the control performance, the computational speeds and the communication burdens of the two approaches are all involved, should be performed.

- **Integration of dynamic orientating of road sections and dynamic routing of traffic flows**

  In Chapter 6, we have developed a novel formulation for the co-optimization problem that jointly determines the orientations of road sections and the routes of traffic flows by assuming a constant travel time on each road section. However, for real applications in traffic networks, the travel time on each road section varies depending on the real-time conditions of traffic in the road section. Actually, the formulation we have developed also works for the case where the travel time on each road section is not fixed. In that case, the dynamic orientating of road sections can be integrated with the dynamic routing of traffic flows.

  Given the discrete nature of the orientating decision variables and the nonlinear dependence of the link travel time on the states of the network, the resulting problem is actually a mixed integer nonlinear programming problem. Although it is computationally hard to solve such a problem, it is recommended that the gain in performance and the increase in computation time should be investigated. Based on the investigation, an approximate solution method, e.g. hierarchical and heuristic control approach, which achieves a well-balanced trade-off between the solution quality and the computation time, may be developed for solving the integrated problem.

- **More efficient multi-agent model predictive control methods for classes of hybrid systems with limited information sharing**

  The multi-agent model predictive control method developed in Chapter 7 has shown high effectiveness in balancing the solution quality, the computation time, and the communication burden. One possible way to further improve the efficiency of the method is to develop a distributed algorithm to compute the global lower bound of the overall objective function, and to combine that algorithm with the search tree building mechanism that has been developed in this thesis. More specifically, in the new overall multi-agent control method, the global lower bound will be computed by the newly developed distributed algorithm while the global upper bound will be computed by the distributed resource allocation coordination algorithm. After that, the bounding technique in the standard branch-and-bound paradigm can be used. Further, the search tree building mechanism developed in this thesis can be used as the branching technique in the standard branch-and-bound paradigm to further explore the solution space.

- **Additional research recommendations**

  The multi-agent control approaches developed in this thesis can be extended to other applications such as:

  - **Control of multi-modal urban transportation networks**

    In Chapters 3 and 4, we have addressed the dynamic routing of a fleet of cybercars in a network that is only open to cybercars. An interesting and challenging research topic based on this work is to consider a network that is open to heterogeneous classes of vehicles, such as cybercars, human-driven cars, buses, trucks, and trams, etc. Actually, in such a network, the

human-driven vehicles may not follow the routes determined by the routing controllers. Therefore, one of the main challenges in addressing the dynamic routing of vehicles in a network with mixed traffic is modeling the behaviors of drivers.

– **Control of baggage handling systems**

  In large and busy airports, there are large-scale baggage handling systems transporting the baggage using destination coded vehicles (DCVs) from the check-in area to the gates and from the gates to the baggage claim area. Given the O-D (origin-destination) demands, the optimal routing of flows of DCVs for these baggage handling systems can be described in a similar fashion as the optimal routing of flows of vehicles in urban transportation systems. Therefore, the methods similar to the routing methods proposed for traffic flows in urban transportation networks in this thesis can be used for routing of flows of DCVs in baggage handling systems.

– **Management of smart electricity grids**

  Electricity grids can be very complex. Therefore, control of these grids, where the optimal power flows are computed so that the overall performance of the network is maximized, is really challenging. In particular, due to the increasing penetration of renewable energy (e.g., wind, solar, tides, and geothermal heat) in electricity generation and the varying availability of those energy sources, control of a large electricity grid involves challenging tasks including predicting the profiles of availability of renewable energy in different regions, predicting the profiles of electricity demands in different regions, and optimizing the electricity generation and distribution.

Appendix A

Properties of Algorithm 7.1

This addendum contains the proof of the optimality conditions and the proof of the oscillation detecting conditions for discrete optimization variables when Algorithm 7.1 presented in Chapter 7 is applied to optimization problems with discrete optimization variables. More specifically, we consider the following optimization problem:

$$\begin{aligned} \min_{u} \ & \sum_{i=1}^{N} f_i(u_i) \\ \text{subject to } \ & u_i \in D_i \\ & \sum_{i=1}^{N} g_i(u_i) \le r \end{aligned} \tag{A.1}$$

where u_i is a scalar, $f_i(\cdot)$ is a convex function, $g_i(\cdot)$ is a monotonically strictly increasing function, and D_i is a finite discrete set. We will explain

- Whether the global optimum is found if none of the optimization variables oscillates;
- How to detect oscillations of discrete optimization variables

when Algorithm 7.1 presented in Chapter 7 is directly applied to (A.1).

A.1 Optimality conditions

First, we define

- if $f_i(\cdot)$ is strictly convex, then

$$u_i^{\text{best}} = \arg\min_{u_i \in D_i} f_i(u_i)$$

- If $f_i(\cdot)$ is not strictly convex, then

$$\begin{aligned} f_i^{\text{best}} &= \min_{u_i \in D_i} f_i(u_i) \\ u_i^{\text{best}} &= \min_{u_i \in D_i,\ f_i(u_i) = f_i^{\text{best}}} u_i \end{aligned}$$

After that, letting $\sum_{i=1}^{N} \theta_i = r$, we decompose the overall problem (A.1) into N subproblems:

- if $f_i(\cdot)$ is strictly convex, then subproblem i is defined by

$$\begin{aligned} \min_{u_i} \ & f_i(u_i) \\ \text{subject to} \ & u_i \in D_i \\ & g_i(u_i) \leq \theta_i \end{aligned}$$

- if $f_i(\cdot)$ is not strictly convex, then subproblem i is defined by

$$\begin{aligned} \min_{u_i} \ & f_i(u_i) \\ \text{subject to} \ & u_i \in D_i \\ & g_i(u_i) \leq \theta_i \\ & u_i \leq u_i^{\text{best}} \end{aligned}$$

Further, based on the definition of u_i^{best}, the following three cases can occur:

Case 1: $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) < r$

Case 2: $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) = r$

Case 3: $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) > r$

Before applying Algorithm 7.1 to the three cases, we want to stress for any subproblem that, given θ_i,

- If $\theta_i \geq g_i(u_i^{\text{best}})$, then constraint $g_i(u_i) \leq \theta_i$ does not pose any restriction. Therefore, $u_i^* = u_i^{\text{best}}$ and[1] $\lambda_i = 0$.
- If $\theta_i < g_i(u_i^{\text{best}})$, then $u_i^* < u_i^{\text{best}}$ and $\lambda_i = -\frac{f_i'(u_i^*)}{g_i'(u_i^*)} > 0$ with $f_i'(u_i^*) < 0$ since $f_i(u_i^*) > f_i(u_i^{\text{best}})$ and $f_i(\cdot)$ is convex, and with $g_i'(u_i^*) > 0$ since $g_i(u_i^*)$ is a monotonically strictly increasing function.

where u_i^* is the solution of subproblem i with θ_i given and λ_i is the Lagrange multiplier corresponding to the constraint $g_i(u_i) \leq \theta_i$ in the subproblem.

Note that if Algorithm 7.1 is applied, the resource allocation at each iteration step z is updated by

$$\theta_i^{(z+1)} = \theta_i^{(z)} + \xi^{(z)} \left(\lambda_i^{(z)} - \frac{1}{N} \sum_{j=1}^{N} \lambda_j^{(z)} \right) \tag{A.2}$$

with

$$\xi^{(z)} > 0, \ \lim_{z \to +\infty} \xi^{(z)} = 0, \ \sum_{z=1}^{+\infty} \xi^{(z)} = +\infty, \ \sum_{z=1}^{+\infty} (\xi^{(z)})^2 < +\infty$$

In the remaining of this section, we let $N = 2$ and prove some properties of the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$ when Algorithm 7.1 is applied to the problem (A.1) in the aforementioned three cases. In Section A.3, we will prove the properties of Algorithm 7.1 for $N > 2$.

[1] When $\theta_i = g_i(u_i^{\text{best}})$, the corresponding λ_i is free. However, in that case we set it equal to 0 by definition.

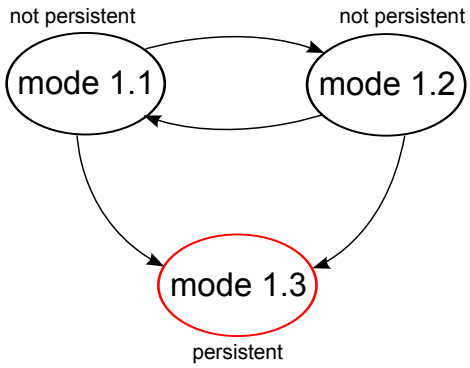

Figure A.1: Mode transition diagram of Case 1 with the final mode marked in red

For Case 1

In this case, $\sum_{i=1}^{2} g_i(u_i^{\text{best}}) < r$. Since $\sum_{i=1}^{2} \theta_i = r$, we consider the following three modes:

- mode 1.1: $\theta_1^{(z)} > g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} < g_2(u_2^{\text{best}})$
- mode 1.2: $\theta_1^{(z)} < g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} > g_2(u_2^{\text{best}})$
- mode 1.3: $\theta_1^{(z)} \geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} \geq g_2(u_2^{\text{best}})$

The mode transition diagram of Case 1 is shown in Figure A.1, of which the proof is given in the following propositions.

Definition: A persistent mode such that once it has been reached, the system stays in that mode.

Proposition 1.1: Let $\delta_2 = \min\limits_{u_2 \in D_2,\ u_2 < u_2^{\text{best}}} -\frac{f_2'(u_2)}{g_2'(u_2)}$; then $\lambda_2^{(z)} \geq \delta_2 > 0$ holds for all z in mode 1.1.

Proof: First, since $-\frac{f_2'(u_2)}{g_2'(u_2)} > 0$ holds for all $u_2 < u_2^{\text{best}}$ and $u_2 \in D_2$, it is directly proved that $\delta_2 > 0$. Further, if mode 1.1 is active at any step z, then $\theta_2^{(z)} < g_2(u_2^{\text{best}})$. Therefore, $u_2^{*,(z)} < u_2^{\text{best}}$. Also note that $u_2^{*,(z)} \in D_2$. Hence, $\lambda_2^{(z)} = -\frac{f_2'\left(u_2^{*,(z)}\right)}{g_2'\left(u_2^{*,(z)}\right)} \geq \delta_2$ always holds in mode 1.1. □

Proposition 1.2: Mode 1.1 is not persistent.
Proof: Let us assume that the system stays in mode 1.1 from some step z_0 on and show that this leads to a contradiction.

If the system is in mode 1.1 at step z_0, then $\theta_1^{(z_0)} > g_1(u_1^{\text{best}})$ and $\theta_2^{(z_0)} < g_2(u_2^{\text{best}})$. In this mode, $u_1^{*,(z_0)} = u_1^{\text{best}}$, $\lambda_1^{(z_0)} = 0$ and $u_2^{*,(z_0)} < u_2^{\text{best}}$, $\lambda_2^{(z_0)} > 0$. According to (A.2), in this mode, at step $z_0 + 1$ we have

$$\theta_1^{(z_0+1)} = \theta_1^{(z_0)} - \frac{1}{2} \cdot \xi^{(z_0)} \cdot \lambda_2^{(z_0)}$$
$$\theta_2^{(z_0+1)} = \theta_2^{(z_0)} + \frac{1}{2} \cdot \xi^{(z_0)} \cdot \lambda_2^{(z_0)}$$

Since $\sum_{z=z_0}^{+\infty}\xi^{(z)}=+\infty$ and $\lambda_2^{(z)}\geq\delta_2>0$ holds for all z in mode 1.1, it is straightforwardly derived that at a certain step z_0+K with $K\geq 1$, either $\theta_1^{(z_0+K)}>g_1(u_1^{\text{best}})$ or $\theta_2^{(z_0+K)}<g_2(u_2^{\text{best}})$ does not hold. This contradicts the condition of mode 1.1. Therefore, mode 1.1 is not persistent and the system will definitely switch to another mode. □

Proposition 1.3: Let $\delta_1=\min\limits_{u_1\in D_1,\ u_1<u_1^{\text{best}}}-\frac{f_1'(u_1)}{g_1'(u_1)}$; then $\lambda_1^{(z)}\geq\delta_1>0$ holds for all z in mode 1.2.

Proof: The proof of this proposition is similar to the one of Proposition 1.1.

Proposition 1.4: Mode 1.2 is not persistent.

Proof: The proof of this proposition is similar to the one of Proposition 1.2.

Proposition 1.5: Mode 1.3 is persistent.

Proof: Suppose the system is in mode 1.3 at step z_0; then we show that the system stay in mode 1.3 for all $z>z_0$. If the system is in mode 1.3 at step z_0, then $\theta_1^{(z_0)}\geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z_0)}\geq g_2(u_2^{\text{best}})$ at step z_0. In this mode, $u_1^{*,(z_0)}=u_1^{\text{best}}$, $\lambda_1^{(z_0)}=0$ and $u_2^{*,(z_0)}=u_2^{\text{best}}$, $\lambda_2^{(z_0)}=0$. According to (A.2), at step z_0+1 we have

$$\theta_1^{(z_0+1)}=\theta_1^{(z_0)}$$
$$\theta_2^{(z_0+1)}=\theta_2^{(z_0)}$$

It is clear that $\theta_1^{(z)}=\theta_1^{(z_0)}$ and $\theta_2^{(z)}=\theta_2^{(z_0)}$ holds for all $z>z_0$. Therefore, mode 1.3 is persistent. Besides, the overall optimal solution $[u_1^{\text{best}}\quad u_2^{\text{best}}]^{\text{T}}$ is directly attained in this mode.

Proposition 1.6: There exists a finite integer $M>0$ such that at any $z\geq M$, mode 1.3 is active i.e., $\theta_1^{(z)}\geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z)}\geq g_2(u_2^{\text{best}})$.

Proof: Let us assume mode 1.3 will never be reached and show that this leads to a contradiction. First, define $\lambda_2^{\max}=\max\limits_{u_2\in D_2,\ u_2<u_2^{\text{best}}}-\frac{f_2'(u_2)}{g_2'(u_2)}$. Since $\lim_{z\to+\infty}\xi^{(z)}=0$, given $\varepsilon=\frac{2\left(r-\sum_{i=1}^{2}g_i(u_i^{\text{best}})\right)}{\lambda_2^{\max}}$, there exists an $M>0$ such that $\xi^{(z)}<\varepsilon$ for all $z\geq M$.

Second, since mode 1.1 and mode 1.2 have been proved not to be persistent, if mode 1.3 will never be reached, there are always mode switches either from mode 1.1 to mode 1.2 or from mode 1.2 to mode 1.1. Assume there is a switch from mode 1.1 to mode 1.2 at some step z_1 with $z_1>M$ (i.e. at step z_1 mode 1.1 is active and at step z_1+1 mode 1.2 is active). Then according to the conditions of mode 1.1 and mode 1.2, we have

$$\theta_1^{(z_1)}>g_1(u_1^{\text{best}}),\ \ \theta_2^{(z_1)}<g_2(u_2^{\text{best}})$$
$$\theta_1^{(z_1+1)}<g_1(u_1^{\text{best}}),\ \ \theta_2^{(z_1+1)}>g_2(u_2^{\text{best}})$$

Hence, we have

$$\theta_1^{(z_1+1)}-\theta_1^{(z_1)}<\theta_1^{(z_1+1)}-g_1(u_1^{\text{best}})<0$$
$$0<\theta_2^{(z_1+1)}-g_2(u_2^{\text{best}})<\theta_2^{(z_1+1)}-\theta_2^{(z)}$$

Since mode 1 is active at step z_1, we have $\theta_1^{(z_1+1)} - \theta_1^{(z_1)} = -\frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{(z_1)}$ and $\theta_2^{(z_1+1)} - \theta_2^{(z_1)} = \frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{(z_1)}$ (see the proof of Proposition 1.1). As a consequence, we have

$$-\frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{(z_1)} < \theta_1^{(z_1+1)} - g_1(u_1^{\text{best}}) < 0$$
$$0 < \theta_2^{(z_1+1)} - g_2(u_2^{\text{best}}) < \frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{(z_1)}$$

So

$$-\frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{(z_1)} < \theta_1^{(z_1+1)} + \theta_2^{(z_1+1)} - g_1(u_1^{\text{best}}) - g_2(u_2^{\text{best}}) < \frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{(z_1)}$$

Since $\theta_2^{(z_1)} < g_2(u_2^{\text{best}})$, $u_2^{*,(z_1)} < u_2^{\text{best}}$. So $\lambda_2^{(z_1)} = -\frac{f_2'\left(u_2^{*,(z_1)}\right)}{g_2'\left(u_2^{*,(z_1)}\right)} < \lambda^{\max}$. Therefore, we have

$$\theta_1^{(z_1+1)} + \theta_2^{(z_1+1)} - g_1(u_1^{\text{best}}) - g_2(u_2^{\text{best}}) < \frac{1}{2} \cdot \xi^{(z_1)} \cdot \lambda_2^{\max}$$

If the switch from mode 1.1 to mode 1.2 happens at $z_1 > M$, we have

$$\theta_1^{(z_1+1)} + \theta_2^{(z_1+1)} - g_1(u_1^{\text{best}}) - g_2(u_2^{\text{best}}) < \frac{1}{2} \cdot \varepsilon \cdot \lambda_2^{\max}$$

and then

$$\theta_1^{(z_1+1)} + \theta_2^{(z_1+1)} - g_1(u_1^{\text{best}}) - g_2(u_2^{\text{best}}) < r - \sum_{i=1}^{2} g_i(u_i^{\text{best}})$$

Since $\theta_1^{(z_1+1)} + \theta_2^{(z_1+1)} = r$, we have

$$g_1(u_1^{\text{best}}) + g_2(u_2^{\text{best}}) > \sum_{i=1}^{2} g_i(u_i^{\text{best}})$$

Clearly, this results in a contradiction since $\sum_{i=1}^{2} g_i(u_i^{\text{best}}) = g_1(u_1^{\text{best}}) + g_2(u_2^{\text{best}})$. Therefore, the assumption that mode 1.3 is never reached does not hold. □

Proposition 1.7: There exists a finite integer $M > 0$ such that the global optimum $[u_1^{\text{best}}\ u_2^{\text{best}}]^{\text{T}}$ is attained at $z = M$.

Proof: According to Proposition 1.6, there exists a finite integer $M > 0$ such that at any $z \geq M$, $\theta_1^{(z)} \geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} \geq g_2(u_2^{\text{best}})$. Therefore, we have $\theta_1^{(M)} \geq g_1^{(M)}(u_1^{\text{best}})$, $\theta_2^{(M)} \geq g_2(u_2^{\text{best}})$ and $u_1^{*,(M)} = u_1^{\text{best}}$, $u_2^{*,(M)} = u_2^{\text{best}}$. Since $f_1(u_1^{\text{best}}) \leq f_1(u_1)$ holds for all $u_1 \in D_1$ and $f_2(u_2^{\text{best}}) \leq f_2(u_2)$ holds for all $u_2 \in D_2$, it is directly derived that $[u_1^{\text{best}}\ u_2^{\text{best}}]^{\text{T}}$ is the global optimum. Finally, since $u_1^{*,(z)} = u_1^{\text{best}}$, $u_2^{*,(z)} = u_2^{\text{best}}$ holds for all $z \geq M$, the global optimum is attained at $z = M$. □

Graph-aided explanation

Based on some figures, in this section we will present a more intuitive explanation for the properties of the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$ when Algorithm 7.1 is applied to problem (A.1) in Case 1. We first arrange each D_i as an ordered set where the elements are ordered in an increasing fashion. After that, for each subproblem i, we define the Lagrange multiplier λ_i

corresponding to $g_i(u_i) \leq \theta_i$ as a function of θ_i and make a plot showing the relationship between the values of λ_i and the values of θ_i. Finally, we present the calculation of $\theta_i^{(z)}$ in Algorithm 7.1 to help explain the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$.

First, we let

$$D_1 = \{u_{1,n_1},\ u_{1,n_1-1},\ ...,\ u_{1,1},\ u_1^{\text{best}},\ ...,\ u_{1,\text{largest}}\}, \qquad n_1 \geq 0$$
$$D_2 = \{u_{2,n_2},\ u_{2,n_2-1},\ ...,\ u_{2,1},\ u_2^{\text{best}},\ ...,\ u_{2,\text{largest}}\}, \qquad n_2 \geq 0$$

with

$$u_{1,n_1} < u_{1,n_1-1} < ... < u_{1,1} < u_1^{\text{best}} < ... < u_{1,\text{largest}}$$
$$u_{2,n_2} < u_{2,n_2-1} < ... < u_{2,1} < u_2^{\text{best}} < ... < u_{2,\text{largest}}$$

Note that $g_i(\cdot)$ a monotonically strictly increasing function. Therefore, we have

$$g_1(u_{1,n_1}) < ... < g_1(u_{1,1}) < g_1(u_1^{\text{best}}) < ... < g_1(u_{1,\text{largest}})$$
$$g_2(u_{2,n_2}) < ... < g_2(u_{2,1}) < g_2(u_2^{\text{best}}) < ... < g_2(u_{2,\text{largest}})$$

Second, given $f_i(\cdot)$, $g_i(\cdot)$, D_i and θ_i, the solution u_i^* to subproblem i is given by

$$u_i^* = \begin{cases} u_{i,n_i}, & \text{if } g_i(u_{i,n_i}) \leq \theta_i < g_i(u_{i,n_i-1}) \\ u_{i,n_i-1}, & \text{if } g_i(u_{i,n_i-1}) \leq \theta_i < g_i(u_{i,n_i-2}) \\ ... & \\ u_{i,1}, & \text{if } g_i(u_{i,1}) \leq \theta_i < g_i(u_i^{\text{best}}) \\ u_i^{\text{best}}, & \text{if } \theta_i \geq g_i(u_i^{\text{best}}) \end{cases} \tag{A.3}$$

Then, the Lagrange multiplier λ_i corresponding to the constraint $g_i(u_i^*) \leq \theta_i$ is given by

$$\lambda_i = \begin{cases} -\frac{f_i'(u_{i,n_i})}{g_i'(u_{i,n_i})}, & \text{if } g_i(u_{i,n_i}) \leq \theta_i < g_i(u_{i,n_i-1}) \\ -\frac{f_i'(u_{i,n_i-1})}{g_i'(u_{i,n_i-1})}, & \text{if } g_i(u_{i,n_i-1}) \leq \theta_i < g_i(u_{i,n_i-2}) \\ ... & \\ -\frac{f_i'(u_{i,1})}{g_i'(u_{i,1})}, & \text{if } g_i(u_{i,1}) \leq \theta_i < g_i(u_i^{\text{best}}) \\ 0, & \text{if } \theta_i \geq g_i(u_i^{\text{best}}) \end{cases} \tag{A.4}$$

Therefore, the dependence of λ_i on θ_i can be expressed as $\lambda_i = \phi_i(\theta_i)$, where ϕ_i is a piecewise constant function. Although the explicit profile of $\phi_i(\cdot)$ depends on $f_i(\cdot)$, $g_i(\cdot)$ and D_i, without loss of generality, representative profiles of $\phi_1(\cdot)$ and $\phi_2(\cdot)$ that describe (A.4) are given in Figure A.2. Since $\theta_1 + \theta_2 = r$, we have

$$\theta_2 = r - \theta_1$$

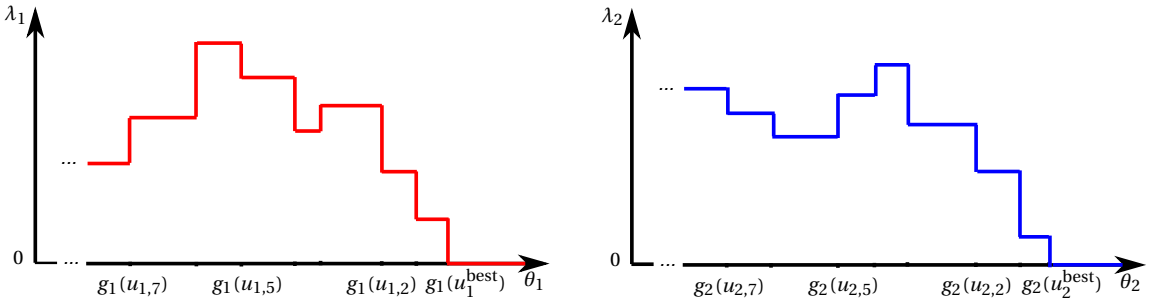

Figure A.2: Representative profiles of $\phi_1(\cdot)$ and $\phi_2(\cdot)$

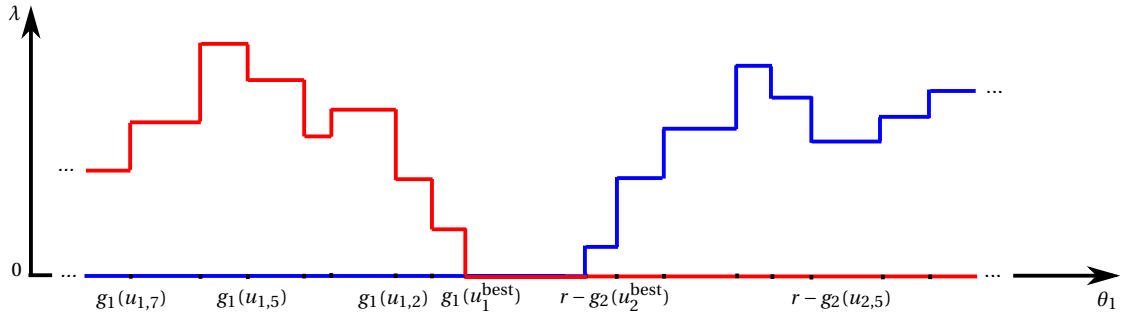

Figure A.3: Representative profiles of $\phi_1(\cdot)$ (in red) and $\phi_3(\cdot)$ (in blue) showing the values of λ_1 and λ_2 along the axis of θ_1 in Case 1

Therefore, given r, the dependence of λ_2 on θ_1 is given by $\lambda_2 = \phi_2(\theta_2) = \phi_2(r-\theta_1) = \phi_3(\theta_1)$. Furthermore, given the profiles of $\phi_1(\cdot)$ and $\phi_2(\cdot)$ shown in Figure A.2 and $g_1(u_1^{\text{best}}) + g_2(u_2^{\text{best}}) < r$ in Case 1, the dependence of λ_1 and λ_2 on θ_1, i.e. the profile of $\phi_1(\cdot)$ and $\phi_3(\cdot)$, is shown in Figure A.3.

Finally, before presenting the calculation of $\theta_1^{(z)}$ in Algorithm 7.1, we divide the axis of θ_1 into three domains:

- domain 1: $\theta_1 < g_1(u_1^{\text{best}})$
- domain 2: $\theta_1 > r - g_2(u_2^{\text{best}})$
- domain 3: $g_1(u_1^{\text{best}}) \le \theta_1 \le r - g_2(u_2^{\text{best}})$

It is seen from Figure A.3 that in domain 1, $\lambda_1 > \lambda_2 = 0$; in domain 2, $\lambda_2 > \lambda_1 = 0$; in domain 3, $\lambda_1 = \lambda_2 = 0$. Note that in Algorithm 7.1, the update of $\theta_1^{(z+1)}$ at each iteration step z is given by

$$\theta_1^{(z+1)} = \theta_1^{(z)} + \xi^{(z)} \frac{\lambda_1^{(z)} - \lambda_2^{(z)}}{2} \tag{A.5}$$

with

$$\xi^{(z)} > 0, \lim_{z\to+\infty} \xi^{(z)} = 0, \sum_{z=1}^{+\infty} \xi^{(z)} = +\infty, \sum_{z=1}^{+\infty} (\xi^{(z)})^2 < +\infty$$

Based on Figure A.3, it can be derived that

- if $\theta_1^{(z_1)}$ is in domain 1, then $\theta_1^{(z_1+1)} = \theta_1^{(z_1)} + \xi^{(z_1)} \frac{\lambda_1^{(z_1)}}{2}$ with $\lambda_1^{(z_1)} \geq \delta_1 > 0$ (see proposition 1.3), which implies that $\theta_1^{(z_1+1)}$ moves towards domain 3. Given that $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$, there exists a finite integer $M_1 \geq z_1$ such that $\theta_1^{(M_1)}$ is in domain 1 and $\theta_1^{(M_1+1)}$ is not in domain 1.

- if $\theta_1^{(z_2)}$ is in domain 2, then $\theta_1^{(z_2+1)} = \theta_1^{(z_2)} - \xi^{(z_2)} \frac{\lambda_2^{(z_2)}}{2}$ with $\lambda_2^{(z_2)} \geq \delta_2 > 0$ (see proposition 1.1), which implies that $\theta_1^{(z_2+1)}$ moves towards domain 3. Given that $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$, there exists a finite integer $M_2 \geq z_2$ such that $\theta_1^{(M_2)}$ is in domain 2 and $\theta_1^{(M_2+1)}$ is not in domain 2.

- Since $\theta_1^{(z)}$ moves towards domain 3 with an increase of $\xi^{(z)} \frac{\lambda_1^{(z)}}{2}$ when it is in domain 1 or with a decrease of $\xi^{(z)} \frac{-\lambda_2^{(z)}}{2}$ when it is in domain 2, given that $\lim_{z \to +\infty} \xi^{(z)} = 0$ and that the width of domain 3 satisfies $|r - g_1(u_1^{\text{best}}) - g_1(u_2^{\text{best}})| > 0$, there exists a finite integer $M > 0$ such that $\theta_1^{(M)}$ reaches domain 3.

- if $\theta_1^{(z)}$ is in domain 3, then $\theta_1^{(z+1)} = \theta_1^{(z)}$, which implies that $\theta_1^{(z)}$ reaches a steady value.

Therefore, in Algorithm 7.1 no matter what is the value of $\theta_1^{(1)}$, $\theta_1^{(z)}$ will reach a point within domain 3 for a finite z and stay at that point afterwards. Since $g_1(u_1^{\text{best}}) \leq \theta_1^{(z)} \leq r - g_2(u_2^{\text{best}})$ in domain 3 and $\theta_1^{(z)} + \theta_2^{(z)} = r$, we have $\theta_2^{(z)} \geq g_2(u_2^{\text{best}})$ in domain 3. More specifically, within domain 3, we have $\theta_1^{(z)} \geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} \geq g_2(u_2^{\text{best}})$. Moreover, given $\theta_1^{(z)} \geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} \geq g_2(u_2^{\text{best}})$, according to (A.3) we have $u_1^{*,(z)} = u_1^{\text{best}}$ and $u_2^{*,(z)} = u_2^{\text{best}}$ with $[u_1^{\text{best}}\ u_2^{\text{best}}]^{\text{T}}$ being the global optimum.

Remark
In Case 1, "$\theta_1^{(z)}$ is in domain 1" is equivalent to "mode 1.2 is active"; "$\theta_1^{(z)}$ is in domain 2" is equivalent to "mode 1.1 is active"; "$\theta_1^{(z)}$ is in domain 3" is equivalent to "mode 1.3 is active". Therefore, the graph-aided explanation of the evolution of $\theta_1^{(z)}$ presented above is consistent with the mode transition diagram in Case 1, i.e. Figure A.1. Although the mathematical proof of the mode transition diagram in Case 1 has been given in Proposition 1.1 to Proposition 1.7, the explanation of the evolution of $\theta_1^{(z)}$ based on Figure A.3 provides a more intuitive understanding of Figure A.1.

For Case 2

In this case, $\sum_{i=1}^{2} g_i(u_i^{\text{best}}) = r$. Since $\sum_{i=1}^{2} \theta_i = r$, we consider the following three modes:

- mode 2.1: $\theta_1^{(z)} > g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} < g_2(u_2^{\text{best}})$

- mode 2.2: $\theta_1^{(z)} < g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} > g_2(u_2^{\text{best}})$

- mode 2.3: $\theta_1^{(z)} = g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} = g_2(u_2^{\text{best}})$

The mode transition diagram of Case 2 is shown in Figure A.4, of which the proof is given in the following propositions.

Proposition 2.1: Mode 2.1 is not persistent.

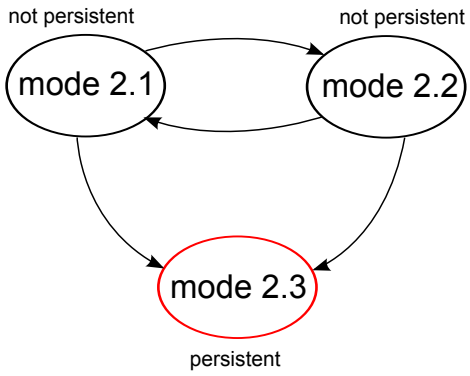

Figure A.4: Mode transition diagram of Case 2 with the final state marked in red

Proof: The proof of this proposition is similar to the one of Proposition 1.2.

Proposition 2.2: Mode 2.2 is not persistent.

Proof: The proof of this proposition is similar to the one of Proposition 1.4.

Proposition 2.3: Mode 2.3 is persistent.

Proof: The proof of this proposition is similar to the one of Proposition 1.5. The overall optimal solution $[u_1^{\text{best}} \;\; u_2^{\text{best}}]^{\text{T}}$ is attained in this mode.

Proposition 2.4: In mode 2.1, $\left|\theta_1^{(z)} - g_1(u_1^{\text{best}})\right|$ and $\left|\theta_2^{(z)} - g_2(u_2^{\text{best}})\right|$ are strictly decreasing as functions of z.

Proof: If mode 2.1 is active at step z_0, we have $\theta_1^{(z_0)} > g_1(u_1^{\text{best}})$ and $\theta_2^{(z_0)} < g_2(u_2^{\text{best}})$ and

$$\theta_1^{(z_0+1)} = \theta_1^{(z_0)} - \frac{1}{2}\cdot\xi^{(z_0)}\cdot\lambda_2^{(z_0)}$$
$$\theta_2^{(z_0+1)} = \theta_2^{(z_0)} + \frac{1}{2}\cdot\xi^{(z_0)}\cdot\lambda_2^{(z_0)}$$

If mode 2.1 is still active at step z_0+1, we have $\theta_1^{(z_0+1)} > g_1(u_1^{\text{best}})$ and $\theta_2^{(z_0+1)} < g_2(u_2^{\text{best}})$. Then we have

$$\begin{aligned}
&\left|\theta_1^{(z_0+1)} - g_1(u_1^{\text{best}})\right| - \left|\theta_1^{(z_0)} - g_1(u_1^{\text{best}})\right| = \theta_1^{(z_0+1)} - g_1(u_1^{\text{best}}) - \left(\theta_1^{(z_0)} - g_1(u_1^{\text{best}})\right)\\
&= \theta_1^{(z_0+1)} - \theta_1^{(z_0)} = -\frac{1}{2}\cdot\xi^{(z_0)}\cdot\lambda_2^{(z_0)} < -\frac{1}{2}\cdot\xi^{(z_0)}\cdot\delta_2 < 0\\
&\left|\theta_2^{(z_0+1)} - g_2(u_2^{\text{best}})\right| - \left|\theta_2^{(z_0)} - g_2(u_2^{\text{best}})\right| = g_2(u_2^{\text{best}}) - \theta_2^{(z_0+1)} - \left(g_2(u_2^{\text{best}}) - \theta_2^{(z_0)}\right)\\
&= \theta_2^{(z_0)} - \theta_2^{(z_0+1)} = -\frac{1}{2}\cdot\xi^{(z_0)}\cdot\lambda_2^{(z_0)} < -\frac{1}{2}\cdot\xi^{(z_0)}\cdot\delta_2 < 0
\end{aligned}$$

Note that δ_2 has been defined in Proposition 1.1. □

Proposition 2.5: In mode 2.2, $\left|\theta_1^{(z)} - g_1(u_1^{\text{best}})\right|$ and $\left|\theta_2^{(z)} - g_2(u_2^{\text{best}})\right|$ are strictly decreasing as functions of z.

Proof: The proof of this proposition is similar to the one of Proposition 2.4.

Proposition 2.6: Let $\sigma_1 = \max\limits_{u_1 \in D_1,\ u_1 < u_1^{\text{best}}} -\frac{f_1'(u_1)}{g_1'(u_1)}$ and $\sigma_2 = \max\limits_{u_2 \in D_2,\ u_2 < u_2^{\text{best}}} -\frac{f_2'(u_2)}{g_2'(u_2)}$. Given $\sigma^{\max} = \max\{\sigma_1, \sigma_2\}$, $\theta_1^{(1)}$ and $\theta_2^{(1)}$, a large integer K and and a small real number $\epsilon = \frac{\sigma^{\max}}{2}\xi^{(K)}$, there exists a finite integer $M > K$ such that $\left|\theta_1^{(z)} - g_1(u_1^{\text{best}})\right| < \epsilon$ and $\left|\theta_2^{(z)} - g_2(u_2^{\text{best}})\right| < \epsilon$ hold for all $z > M$.

Proof: If there is a switch from mode 2.1 to mode 2.2 at step z_0, we have $\theta_1^{(z_0)} > g_1(u_1^{\text{best}})$ and $\theta_2^{(z_0)} < g_2(u_2^{\text{best}})$ and $\theta_1^{(z_0+1)} < g_1(u_1^{\text{best}})$ and $\theta_2^{(z_0+1)} > g_2(u_2^{\text{best}})$, and also

$$\left|\theta_1^{(z_0)} - g_1(u_1^{\text{best}})\right| = \theta_1^{(z_0)} - g_1(u_1^{\text{best}}) < \theta_1^{(z_0)} - \theta_1^{(z_0+1)} < \frac{1}{2}\cdot\xi^{(z_0)}\lambda_2^{(z_0)} < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$
$$\left|\theta_1^{(z_0+1)} - g_1(u_1^{\text{best}})\right| = g_1(u_1^{\text{best}}) - \theta_1^{(z_0+1)} < \theta_1^{(z_0)} - \theta_1^{(z_0+1)} < \frac{1}{2}\cdot\xi^{(z_0)}\lambda_2^{(z_0)} < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$
$$\left|\theta_2^{(z_0)} - g_2(u_2^{\text{best}})\right| = g_2(u_2^{\text{best}}) - \theta_2^{(z_0)} < \theta_2^{(z_0+1)} - \theta_2^{(z_0)} < \frac{1}{2}\cdot\xi^{(z_0)}\lambda_2^{(z_0)} < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$
$$\left|\theta_2^{(z_0+1)} - g_2(u_2^{\text{best}})\right| = \theta_2^{(z_0+1)} - g_2(u_2^{\text{best}}) < \theta_2^{(z_0+1)} - \theta_2^{(z_0)} < \frac{1}{2}\cdot\xi^{(z_0)}\lambda_2^{(z_0)} < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$

Likewise, if there is a switch from mode 2.2 to mode 2.1 at step z_0, we have

$$\left|\theta_1^{(z_0)} - g_1(u_1^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$
$$\left|\theta_1^{(z_0+1)} - g_1(u_1^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$
$$\left|\theta_2^{(z_0)} - g_2(u_2^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$
$$\left|\theta_2^{(z_0+1)} - g_2(u_2^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(z_0)}\sigma^{\max}$$

If mode 2.3 is not reached for any $z < \infty$, then there are repeated mode transitions between mode 2.1 and 2.2 since neither mode 2.1 nor 2.2 is persistent. Therefore, no matter what is the value of K, there exists a finite integer $M > K$ such that a mode switch (no matter it is from mode 2.1 to mode 2.2 or from mode 2.2 to mode 2.1) occurs at step M. Hence, we have

$$\left|\theta_1^{(M)} - g_1(u_1^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(M)}\sigma^{\max} < \frac{1}{2}\cdot\xi^{(K)}\sigma^{\max} = \epsilon$$
$$\left|\theta_1^{(M+1)} - g_1(u_1^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(M)}\sigma^{\max} < \frac{1}{2}\cdot\xi^{(K)}\sigma^{\max} = \epsilon$$
$$\left|\theta_2^{(M)} - g_2(u_2^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(M)}\sigma^{\max} < \frac{1}{2}\cdot\xi^{(K)}\sigma^{\max} = \epsilon$$
$$\left|\theta_2^{(M+1)} - g_2(u_2^{\text{best}})\right| < \frac{1}{2}\cdot\xi^{(M)}\sigma^{\max} < \frac{1}{2}\cdot\xi^{(K)}\sigma^{\max} = \epsilon$$

Since we have also proved in Proposition 2.4 and 2.5 that $\left|\theta_1^{(z)} - g_1(u_1^{\text{best}})\right|$ and $\left|\theta_2^{(z)} - g_2(u_2^{\text{best}})\right|$ are strictly decreasing in mode 2.1 and 2.2, we can conclude that at any step $z > M$, $\left|\theta_1^{(z)} - g_1(u_1^{\text{best}})\right| < \epsilon$ and $\left|\theta_2^{(z)} - g_2(u_2^{\text{best}})\right| < \epsilon$ always hold.

Finally, if mode 2.3 is reached at $z_1 < \infty$, no matter what is the value of K, there exists a finite integer $M > K$ such that $\left|\theta_1^{(z)} - g_1(u_1^{\text{best}})\right| < \epsilon$ and $\left|\theta_2^{(z)} - g_2(u_2^{\text{best}})\right| < \epsilon$ hold for all $z > M$. More specifically, if $z_1 < K$, then for any $z > z_1$, we have $|\theta_1^{(z)} - g_1(u_1^{\text{best}})| = 0 < \epsilon$ and $|\theta_2^{(z)} - g_2(u_2^{\text{best}})| = 0 < \epsilon$. Then, by letting $M = K + 1 > z_1$, for any $z > M$, we have

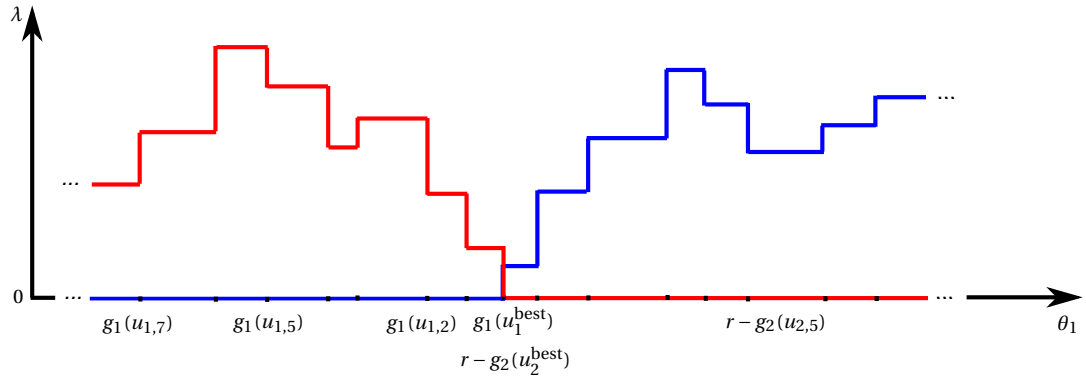

Figure A.5: Representative profiles of $\phi_1(\cdot)$ (in red) and $\phi_3(\cdot)$ (in blue) showing the values of λ_1 and λ_2 along the axis of θ_1 in Case 2

$|\theta_1^{(z)} - g_1(u_1^{\text{best}})| = 0 < \epsilon$ and $|\theta_2^{(z)} - g_2(u_2^{\text{best}})| = 0 < \epsilon$. If $z_1 > K$, by letting $M = z_1$, for all $z > M$, we have $|\theta_1^{(z)} - g_1(u_1^{\text{best}})| = 0 < \epsilon$ and $|\theta_2^{(z)} - g_2(u_2^{\text{best}})| = 0 < \epsilon$. □

Proposition 2.7: $\lim_{z\to\infty} \theta_1^{(z)} = g_1(u_1^{\text{best}})$ and $\lim_{z\to\infty} \theta_2^{(z)} = g_1(u_2^{\text{best}})$.
Proof: The proof of this proposition can be directly derived from Proposition 2.6 with $K = \infty$.

Graph-aided explanation

As for Case 1, we also present a more intuitive explanation for the properties of the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$ when Algorithm 7.1 is applied to problem (A.1) in Case 2. Note that the procedure of explanation in this case is the same as that in Case 1. Therefore, for this case, we skip the parts that are identical to those in Case 1, and we only present the parts that are different from those in Case 1.

First, given the profiles of $\phi_1(\cdot)$ and $\phi_2(\cdot)$ shown in Figure A.2 and $g_1(u_1^{\text{best}}) + g_2(u_2^{\text{best}}) = r$ in Case 2, the dependence of λ_1 and λ_2 on θ_1, i.e. the profile of $\phi_1(\cdot)$ and $\phi_3(\cdot)$, is shown in Figure A.5.

Next, we divide the axis of θ_1 into three domains:

- domain 1: $\theta_1 < g_1(u_1^{\text{best}})$
- domain 2: $\theta_1 > g_1(u_1^{\text{best}})$
- domain 3: $\theta_1 = g_1(u_1^{\text{best}})$

It is seen from Figure A.5 that in domain 1, $\lambda_1 > \lambda_2 = 0$; in domain 2, $\lambda_2 > \lambda_1 = 0$; in domain 3, $\lambda_1 = \lambda_2 = 0$. Note that in Algorithm 7.1, the update of $\theta_1^{(z+1)}$ at each iteration step z is given by (A.5). Based on Figure A.5, it can be derived that

- domain 3, which is actually a point on the axis of θ_1, is the equilibrium point of $\theta_1^{(z)}$.

Therefore, in Algorithm 7.1 no matter what is the value of $\theta_1^{(1)}$, as the iteration step z goes to $+\infty$, $\theta_1^{(z)}$ will reach the equilibrium point where $\theta_1^{(z)} = g_1(u_1^{\text{best}})$. Since $\theta_1^{(z)} + \theta_2^{(z)} = r$ and $\sum_{i=1}^{2} g_i(u_i^{\text{best}}) = r$ in Case 2, we have $\theta_2^{(z)} = g_2(u_2^{\text{best}})$ at the equilibrium point. More specifically, at the equilibrium point, we have $\theta_1^{(z)} = g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} = g_2(u_2^{\text{best}})$. Moreover, according to (A.3) we have $u_1^{*,(z)} = u_1^{\text{best}}$ and $u_2^{*,(z)} = u_2^{\text{best}}$ with $[u_1^{\text{best}}\ u_2^{\text{best}}]^{\text{T}}$ being the global optimum.

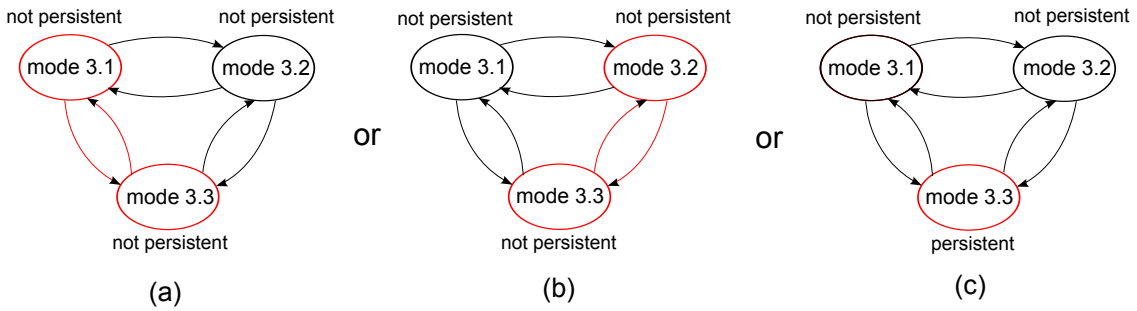

Figure A.6: Mode transition diagram of Case 3 with the final state marked in red

For Case 3

In this case, $\sum_{i=1}^{2} g_i(u_i^{\text{best}}) > r$. Since $\sum_{i=1}^{2} \theta_i = r$, we consider the following three modes:

- mode 3.1: $\theta_1^{(z)} \geq g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} < g_2(u_2^{\text{best}})$
- mode 3.2: $\theta_1^{(z)} < g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} \geq g_2(u_2^{\text{best}})$
- mode 3.3: $\theta_1^{(z)} < g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} < g_2(u_2^{\text{best}})$

The mode transition diagram of Case 3 is shown in Figure A.6, of which the proof is given in the following propositions.

Proposition 3.1: Mode 3.1 is not persistent.
Proof: The proof of this proposition is similar to the one of Proposition 1.2.

Proposition 3.2: Mode 3.2 is not persistent.
Proof: The proof of this proposition is similar to the one of Proposition 1.4.

Proposition 3.3: Given $\frac{f_1'(u_1)}{g_1'(u_1)} \neq \frac{f_2'(u_2)}{g_2'(u_2)}$ for all $u_1 \in D_1$ and $u_2 \in D_2$ with $u_1 < u_1^{\text{best}}$ and $u_2 < u_2^{\text{best}}$, then either $u_1^{*,(z)}$ or $u_2^{*,(z)}$ will not reach a stable value.
Proof: Since mode 3.1 is not persistent, given mode 3.1 is active at some step $z_1 > 0$ with $u_1^{*,(z_1)} = u_1^{\text{best}}$, there will be a switch from mode 3.1 to either mode 3.2 or mode 3.3. Assume the switch happens at step z_2 with $z_2 > z_1$; then we have $u_1^{*,(z_2)} < u_1^{\text{best}}$. Therefore, $u_1^{*,(z_2)} \neq u_1^{*,(z_1)}$.

Since mode 3.2 is not persistent, given mode 3.2 is active at some step $z_3 > 0$ with $u_2^{*,(z_3)} = u_2^{\text{best}}$, there will be a switch from mode 3.2 to either mode 3.1 or mode 3.3. Assume the switch happens at step z_4 with $z_4 > z_3$, we have $u_2^{*,(z_4)} < u_2^{\text{best}}$. Therefore, $u_2^{*,(z_4)} \neq u_2^{*,(z_3)}$.

Depending on whether mode 3.3 is persistent, two situations are considered. First, if mode 3.3 is not persistent, given mode 3.3 is active at some step z_5 with $u_1^{*,(z_5)} < u_1^{\text{best}}$ and $u_2^{*,(z_5)} < u_2^{\text{best}}$, there will be a switch from mode 3.3 to either mode 3.1 or mode 3.2. If the switch is from mode 3.3 to mode 3.1 and happens at step z_6 with $z_6 > z_5$, we have $u_1^{*,(z_6)} = u_1^{\text{best}}$ and then $u_1^{*,(z_6)} \neq u_1^{*,(z_5)}$. If the switch is from mode 3.3 to mode 3.2 and happens at z_7 with $z_7 > z_5$, we have $u_2^{*,(z_7)} = u_2^{\text{best}}$ and then $u_2^{*,(z_7)} \neq u_2^{*,(z_5)}$.

Second, if mode 3.3 is persistent, given mode 3.3 is active at step some step z_8, we have $u_1^{*,(z_8)} < u_1^{\text{best}}$, $\lambda_1^{(z_8)} > 0$ and $u_2^{*,(z_8)} < u_2^{\text{best}}$, $\lambda_2^{(z_8)} > 0$. According to (A.2), in this mode, $\theta_1^{(z_8+1)}$

and $\theta_2^{(z_8+1)}$ are given by

$$\theta_1^{(z_8+1)} = \theta_1^{(z_8)} - \frac{\lambda_2^{(z_8)} - \lambda_1^{(z_8)}}{2} \cdot \xi^{(z_8)}$$
$$\theta_2^{(z_8+1)} = \theta_2^{(z_8)} + \frac{\lambda_2^{(z_8)} - \lambda_1^{(z_8)}}{2} \cdot \xi^{(z_8)}$$

Since $\lambda_1^{(z_8)} = -\frac{f_1'\left(u_1^{*,(z_8)}\right)}{g_1'\left(u_1^{*,(z_8)}\right)}$, $\lambda_2^{(z_8)} = -\frac{f_2'\left(u_2^{*,(z_8)}\right)}{g_2'\left(u_2^{*,(z_8)}\right)}$ and $\frac{f_1'\left(u_1^{*,(z_8)}\right)}{g_1'\left(u_1^{*,(z_8)}\right)} \neq \frac{f_2'\left(u_2^{*,(z_8)}\right)}{g_2'\left(u_2^{*,(z_8)}\right)}$, we have $\lambda_1^{(z_8)} \neq \lambda_2^{(z_8)}$. Therefore, also because $\sum_{z=z_8}^{+\infty} \xi^{(z)} = +\infty$ and $\lambda_1^{(z_8+j)} \neq \lambda_2^{(z_8+j)}$ with $j \geq 0$, $\theta_1^{(z_8+j)}$ keeps increasing (or decreasing) and $\theta_2^{(z_8+j)}$ keeps decreasing (or increasing) until at step z_9 with $z_9 > z_8$ either $u_1^{*,(z_9)} \neq u_1^{*,(z_8)}$ or $u_2^{*,(z_9)} \neq u_2^{*,(z_8)}$. □

Proposition 3.4: Given $\frac{f_1'(u_1)}{g_1'(u_1)} \neq \frac{f_2'(u_2)}{g_2'(u_2)}$ for all $u_1 \in D_1$ and $u_2 \in D_2$ with $u_1 < u_1^{\text{best}}$ and $u_2 < u_2^{\text{best}}$, depending on the value of r, the mode transition diagram of Case 3 can be any of the three kinds shown in Figure A.6.
Proof: The proof will be given using the graph-aided explanation.

Graph-aided explanation

As for Case 1 and Case 2, based on some figures, we present a more intuitive explanation for the properties of the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$ when Algorithm 7.1 is applied to problem (A.1) in Case 3. The procedure of explanation in this case is the same as that in Case 1. Therefore, for this case, we skip the parts that are identical to those in Case 1, and we only present the parts that are different from those in Case 1.

First, since $g_1(u_1^{\text{best}}) + g_2(u_2^{\text{best}}) < r$ in Case 3, the profiles of $\phi_1(\cdot)$ and $\phi_3(\cdot)$ along the axis of θ_1 will intersect. Depending on different values of r and different profiles of $\phi_1(\cdot)$ and $\phi_2(\cdot)$, $\phi_1(\cdot)$ and $\phi_3(\cdot)$ may intersect in various ways. Two representative examples are shown in Figures A.7 and A.8 for the profiles of $\phi_1(\cdot)$ and $\phi_2(\cdot)$ shown in Figure A.2 where we consider representative values r_x (for Figure A.7) and r_y (for Figure A.8) of r with $r_x > r_y$. We call the case of intersection of $\phi_1(\cdot)$ and $\phi_3(\cdot)$ shown in Figure A.7 subcase 3.x and we call the one shown in Figure A.8 subcase 3.y. In the following, we will explain the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$ in subcase 3.x and in subcase 3.y. Note that the explanation of the evolution of $\theta_i^{(z)}$ and $u_i^{*,(z)}$ in other cases of intersection of $\phi_1(\cdot)$ and $\phi_3(\cdot)$ can be done in a similar way.

For subcase 3.x

Given the dependence of λ_1 and λ_2 on θ_1 shown in Figure A.7, we divide the axis of θ_1 into two domains:

- domain 1: $\theta_1 < g_1(u_1^{\text{best}})$
- domain 2: $\theta_1 \geq g_1(u_1^{\text{best}})$

It is seen from Figure A.7 that in domain 1, $\lambda_1 > \lambda_2$; in domain 2, $\lambda_2 > \lambda_1$. Note that in Algorithm 7.1, the update of $\theta_1^{(z+1)}$ at each iteration step z is given by (A.5). Based on Figure A.7, it can be derived that

- if $\theta_1^{(z_1)}$ is in domain 1, then $\theta_1^{(z_1+1)} > \theta_1^{(z_1)}$, which implies that $\theta_1^{(z_1+1)}$ moves towards

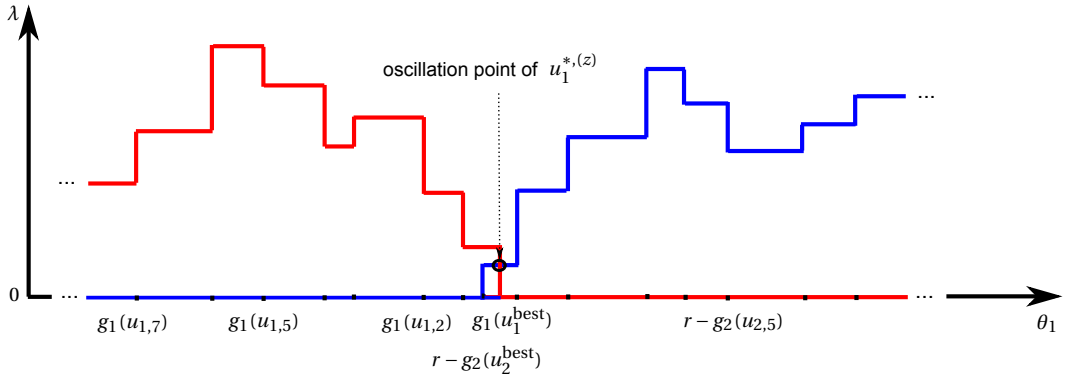

Figure A.7: Representative profiles of $\phi_1(\cdot)$ (in red) and $\phi_3(\cdot)$ (in blue) showing the values of λ_1 and λ_2 along the axis of θ_1 in subcase 3.x

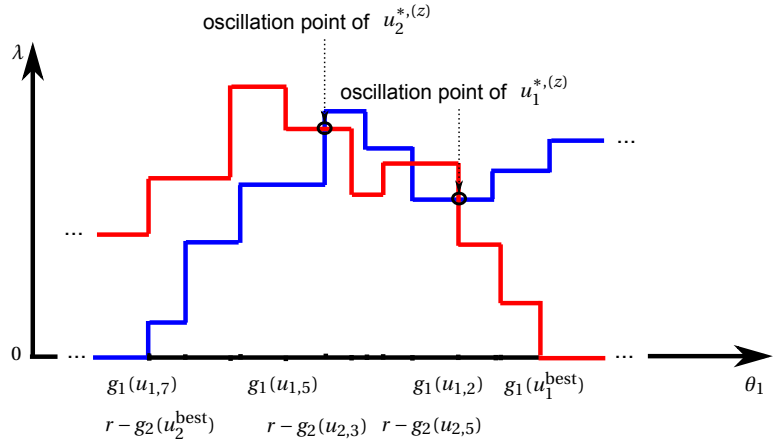

Figure A.8: Representative profiles of $\phi_1(\cdot)$ (in red) and $\phi_3(\cdot)$ (in blue) showing the values of λ_1 and λ_2 along the axis of θ_1 in subcase 3.y

domain 2. Given that $\sum_{z=1}^{+\infty}\xi^{(z)} = +\infty$, there exists a finite integer $M_1 \geq z_1$ such that $\theta_1^{(M_1)}$ is in domain 1 and $\theta_1^{(M_1+1)}$ is in domain 2.

- if $\theta_1^{(z_2)}$ is in domain 2, then $\theta_1^{(z_2+1)} < \theta_1^{(z_2)}$, which implies that $\theta_1^{(z_2+1)}$ moves towards domain 1. Given that $\sum_{z=1}^{+\infty}\xi^{(z)} = +\infty$, there exists a finite integer $M_2 \geq z_2$ such that $\theta_1^{(M_2)}$ is in domain 2 and $\theta_1^{(M_2+1)}$ in domain 1.

- for ϵ an arbitrarily small positive number, given that $\lim_{z\to+\infty}\xi^{(z)} = 0$, there exists a finite integer $M > 0$ such that $|\theta_1^{(z)} - g_1(u_1^{\text{best}})| < \epsilon$ holds for all $z \geq M$, which implies that $\theta_1^{(z)}$ will converge to a small neighborhood of the point $\theta_1 = g_1(u_1^{\text{best}})$; moreover, the value of $u_1^{*,(z)}$ oscillates between $u_{1,1}$ and u_1^{best} as a function of z after $\theta_1^{(z)}$ reaches the small neighborhood $|\theta_1^{(z)} - g_1(u_1^{\text{best}})| < \epsilon$.

- the oscillation of the value of $u_1^{*,(z)}$ as a function of z is characterized by the jumps of the value of $u_1^{*,(z)}$ between $u_{1,1}$ and u_1^{best} at steps $z_{\text{jump},j}$ with j denoting the index of the jump. Each jump of the value of $u_1^{*,(z)}$ at a step $z_{\text{jump},j}$ is characterized by

$$u_1^{*,(z_{\text{jump},j}+1)} \neq u_1^{*,(z_{\text{jump},j})}$$
$$\text{sgn}\left(\Delta\theta_1^{(z_{\text{jump},j}+1)}\right) \neq \text{sgn}\left(\Delta\theta_1^{(z_{\text{jump},j})}\right)$$

with $\Delta\theta_1^{(z_{\text{jump},j})} = \theta_1^{(z_{\text{jump},j})} - \theta_1^{(z_{\text{jump},j}-1)}$.

Note that after $\theta_1^{(z)}$ reaches the small neighborhood $|\theta_1^{(z)} - g_1(u_1^{\text{best}})| < \epsilon$, "$\theta_1^{(z)}$ is in domain 1" is equivalent to "mode 3.3 is active"; "$\theta_1^{(z)}$ is in domain 2" is equivalent to "mode 3.1 is active". Therefore, the graph-aided explanation of evolution of $\theta_1^{(z)}$ presented above is consistent with the mode transition diagram indicated by (a) in Figure A.6.

Remark
If we swap θ_1 and θ_2, u_1 and u_2, and λ_1 and λ_2, then the graph-aided explanation presented above describes the evolution of $\theta_2^{(z)}$, which is consistent with the mode transition diagram indicated by (b) in in Figure A.6.

For subcase 3.y
Given the dependence of λ_1 and λ_2 on θ_1 shown in Figure A.8, we divide the axis of θ_1 into four domains:

- domain 1: $\theta_1 \leq r - g_2(u_{2,3})$
- domain 2: $r - g_2(u_{2,3}) < \theta_1 \leq r - g_2(u_{2,5})$
- domain 3: $r - g_2(u_{2,5}) < \theta_1 < g_1(u_{1,2})$
- domain 4: $\theta_1 \geq g_1(u_{1,2})$

It is seen from Figure A.8 that in domain 1, $\lambda_1 > \lambda_2$; in domain 2, $\lambda_2 > \lambda_1$; in domain 3, $\lambda_1 > \lambda_2$; in domain 4, $\lambda_2 > \lambda_1$. Note that in Algorithm 7.1, the update of $\theta_1^{(z+1)}$ at each iteration step z is given by (A.5). Based on Figure A.8, it can be derived that

- if $\theta_1^{(z_1)}$ is in domain 1, then $\theta_1^{(z_1+1)} > \theta_1^{(z_1)}$, which implies that $\theta_1^{(z_1+1)}$ moves towards domain 2. Given that $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$, there exists a finite integer $M_1 \geq z_1$ such that $\theta_1^{(M_1)}$ is in domain 1 and $\theta_1^{(M_1+1)}$ is not in domain 1.

- if $\theta_1^{(z_2)}$ is in domain 2, then $\theta_1^{(z_2+1)} < \theta_1^{(z_2)}$, which implies that $\theta_1^{(z_2+1)}$ moves towards domain 1. Given that $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$, there exists a finite integer $M_2 \geq z_2$ such that $\theta_1^{(M_2)}$ is in domain 2 and $\theta_1^{(M_2+1)}$ is not in domain 2.

- if $\theta_1^{(z_3)}$ is in domain 3, then $\theta_1^{(z_3+1)} > \theta_1^{(z_3)}$, which implies that $\theta_1^{(z_3+1)}$ moves towards domain 4. Given that $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$, there exists a finite integer $M_3 \geq z_3$ such that $\theta_1^{(M_3)}$ is in domain 3 and $\theta_1^{(M_3+1)}$ is not in domain 3.

- if $\theta_1^{(z_4)}$ is in domain 4, then $\theta_1^{(z_4+1)} < \theta_1^{(z_4)}$, which implies that $\theta_1^{(z_4+1)}$ moves towards domain 3. Given that $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$, there exists a finite integer $M_4 \geq z_4$ such that $\theta_1^{(M_4)}$ is in domain 4 and $\theta_1^{(M_4+1)}$ is not in domain 4.

- for ϵ an arbitrarily small positive number, given that $\lim_{z\to+\infty} \xi^{(z)} = 0$, depending on the value of $\theta_1^{(1)}$, the convergence of $\theta_1^{(z)}$ will be either

  – there exists a finite integer $M > 0$ such that $\left|\theta_1^{(z)} - \left(r - g_2(u_{2,3})\right)\right| < \epsilon$ holds for all $z \geq M$, which implies that $\theta_1^{(z)}$ will converge to a small neighborhood of the point $\theta_1 = r - g_2(u_{2,3})$.

  or

  – there exists a finite integer $M > 0$ such that $\left|\theta_1^{(z)} - g_1(u_{1,2})\right| < \epsilon$ holds for all $z \geq M$, which implies that $\theta_1^{(z)}$ will converge to a small neighborhood of the point $\theta_1 = g_1(u_{1,2})$.

- if $\theta_1^{(z)}$ converges to the small neighborhood $\left|\theta_1^{(z)} - \left(r - g_2(u_{2,3})\right)\right| < \epsilon$, then the value of $u_2^{*,(z)}$ oscillates between $u_{2,3}$ and $u_{2,4}$ as a function of z after $\theta_1^{(z)}$ reaches the small neighborhood. The oscillation of the value of $u_2^{*,(z)}$ as a function of z is characterized by the jumps of the value $u_2^{*,(z)}$ between $u_{2,3}$ and $u_{2,4}$ at steps $z_{\text{jump},j}$. Each jump of the value of $u_2^{*,(z)}$ at a step $z_{\text{jump},j}$ is characterized by

$$u_2^{*,(z_{\text{jump},j}+1)} \neq u_2^{*,(z_{\text{jump},j})}$$
$$\text{sgn}\left(\Delta\theta_2^{(z_{\text{jump},j}+1)}\right) \neq \text{sgn}\left(\Delta\theta_2^{(z_{\text{jump},j})}\right)$$

  with $\Delta\theta_2^{(z_{\text{jump},j})} = \theta_2^{(z_{\text{jump},j})} - \theta_2^{(z_{\text{jump},j}-1)}$.

- if $\theta_1^{(z_{\text{jump},j})}$ converges to the small neighborhood $\left|\theta_1^{(z)} - g_1(u_{1,2})\right| < \epsilon$, then the value of $u_1^{*,(z)}$ oscillates between $u_{1,2}$ and $u_{1,3}$ as a function of z after $\theta_1^{(z)}$ reaches the small neighborhood. The oscillation of the value of $u_1^{*,(z)}$ as a function of z is characterized by the jumps of the value of $u_1^{*,(z)}$ between $u_{1,2}$ and $u_{1,3}$ at steps $z_{\text{jump},j}$. Each jump of

the value of $u_1^{*,(z)}$ at a step $z_{\text{jump},j}$ is characterized by

$$u_1^{*,(z_{\text{jump},j}+1)} \neq u_1^{*,(z_{\text{jump},j})}$$
$$\text{sgn}\left(\Delta\theta_1^{(z_{\text{jump},j}+1)}\right) \neq \text{sgn}\left(\Delta\theta_1^{(z_{\text{jump},j})}\right)$$

with $\Delta\theta_1^{(z_{\text{jump},j})} = \theta_1^{(z_{\text{jump},j})} - \theta_1^{(z_{\text{jump},j}-1)}$.

Note that $\theta_1^{(z)} < g_1(u_1^{\text{best}})$ and $\theta_2^{(z)} < g_2(u_2^{\text{best}})$ hold within $\left|\theta_1^{(z)} - \left(r - g_2(u_{2,3})\right)\right| < \epsilon$ and within $\left|\theta_1^{(z)} - g_1(u_{1,2})\right| < \epsilon$. Therefore, no matter which small neighborhood $\theta_1^{(z)}$ eventually converges to, the graph-aided explanation of evolution of $\theta_1^{(z)}$ presented above is consistent with the mode transition diagram indicated by (c) in Figure A.6.

A.2 Oscillation detection of a discrete optimization variable

Proposition 4.1: The oscillation of the value of a discrete optimization variable u_i can be determined by detecting $u_i^{*,(z+1)} \neq u_i^{*,(z)}$ and $\text{sgn}\left(\Delta\theta_i^{(z+1)}\right) \neq \text{sgn}\left(\Delta\theta_i^{(z)}\right)$.
Proof: The proof of this proposition has been given in the graph-aided explanation for Case 3 in Section A.1, especially in the discussion of subcase 3.1 and subcase 3.2.

A.3 General properties

In the previous sections, we have proved some properties of applying Algorithm 7.1 presented in Chapter 7 to the problem (A.1) with $N = 2$ for Case 1 where $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) < r$, for Case 2 where $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) = r$, and for Case 3 where $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) < r$. In this section, we prove that the properties for Case 1 and for Case 2 still hold when Algorithm 7.1 is applied to the problem for any $N > 2$. Note that for Case 3, when $N > 2$, the explanation for the properties of applying Algorithm 7.1 to (A.1) is very complicated, especially the explanation for the oscillation of the discrete optimization variables. For the sake of simplicity, we do not present the explanation for the properties of applying Algorithm 7.1 to (A.1) for Case 3 with $N > 2$.

We define $I_1^{(z)} = \left\{i | \theta_i \geq g_i(u_i^{\text{best}})\right\}$, $I_2^{(z)} = \left\{j | \theta_j < g_j(u_j^{\text{best}})\right\}$, and $I = \{1, 2, ..., N\}$. It is obvious that $I = I_1^{(z)} \cup I_2^{(z)}$ holds for all iteration step z.

For Case 1

Proposition 5.1: In the case $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) < r$, there exists a finite integer $M \geq 0$ such that at all steps $z \geq M$, $I_2^{(z)} = \emptyset$.
Proof: In this case, with $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) < r$, we want to prove that no matter what the values of $\theta_i^{(1)}$ for $i \in I$ are, $\theta_i^{(z)}$ will eventually reach a steady state with $\theta_i^{(z)} \geq g_i(u_i^{\text{best}})$ for all i. In order to prove I_2 will eventually be empty, we first assume that I_2 will never be empty and then find a contradiction.

At any step $z > 0$, we have $\lambda_i^{(z)} = 0$ for all $i \in I_1^{(z)}$ and $\lambda_j^{(z)} > 0$ for all $j \in I_2^{(z)}$. Now let us

define

$$\bar{\lambda}^{(z)} = \frac{1}{N}\Big(\sum_{j\in I_2^{(z)}} \lambda_j^{(z)} + \sum_{i\in I_1^{(z)}} \lambda_i^{(z)} \Big) = \frac{1}{N} \sum_{j\in I_2^{(z)}} \lambda_j^{(z)}$$

If $I_2^{(z)} \neq \emptyset$ at step $z > 0$, we have $\bar{\lambda}^{(z)} > 0$. We define for all $i \in I$

$$\delta_i = \min_{u_i\in D_i,\ u_i<u_i^{\text{best}}} -\frac{f_i^{'}(u_i)}{g_i^{'}(u_i)}$$

Further, we define

$$\delta^{\min} = \min_{i\in I} \delta_i$$

Since $f_i(\cdot)$ is convex and $f_i(u_i^{\text{best}}) < f_i(u_i)$ for $u_i \in D_i$ with $u_i < u_i^{\text{best}}$, we have $f_i'(u_i) < 0$ for $u_i \in D_i$ with $u_i < u_i^{\text{best}}$. In addition, since $g_i(\cdot)$ is monotonically strictly increasing, we have $g_i'(\cdot) > 0$. Therefore, we have $-\frac{f_i^{'}(u_i)}{g_i^{'}(u_i)} > 0$ for $u_i \in D_i$ with $u_i < u_i^{\text{best}}$ and it is directly derived that $\delta_i > 0$ holds for all $i \in I$ and $\delta^{\min} > 0$. Therefore, if $I_2^{(z)} \neq \emptyset$ at step z, then for every $j \in I_2^{(z)}$, we have $u_j^{*,(z)} < u_j^{\text{best}}$ and $\lambda_j^{(z)} = -\frac{f_j^{'}(u_j^{*,(z)})}{g_j^{'}(u_j^{*,(z)})} \geq \delta^{\min}$. So we have

$$\bar{\lambda}^{(z)} \geq \frac{\delta^{\min}}{N} > 0$$

Now let us define

$$\sigma_i = \max_{x_i\in D_i,\ u_i<u_i^{\text{best}}} -\frac{f_i^{'}(u_i)}{g_i^{'}(u_i)}$$

$$\sigma^{\max} = \max_{i\in I} \sigma_i$$

Like δ_i and $\delta_i^{\min}$, $\sigma_i > 0$ holds for all $i \in I$ and $\sigma^{\max} > 0$. Besides, $\sigma^{\max}$ is finite since I and all D_i have finite elements. Therefore, if $I_2^{(z)} \neq \emptyset$ at step z, we have for all $j \in I_2^{(z)}$

$$\lambda_j^{(z)} - \bar{\lambda}^{(z)} < \lambda_j^{(z)} \leq \sigma^{\max}$$

Now let us define the nonnegative function

$$J(z) = \sum_{i=1}^{N} \left(\theta_i^{(z)} - g_i(u_i^{\text{best}})\right)^2$$

Then

$$J(z+1) - J(z) = \sum_{i=1}^{N} \left(\theta_i^{(z+1)} - \theta_i^{(z)}\right)\left(\theta_i^{(z+1)} + \theta_i^{(z)} - 2g_i(u_i^{\text{best}})\right)$$

At any step z, if $I_2^{(z)} \neq \emptyset$, we have

$$\begin{aligned}
&J(z+1) - J(z) \\
&= \sum_{i \in I_1^{(z)}} -\xi^{(z)} \bar{\lambda}^{(z)} \left(\theta_i^{(z+1)} + \theta_i^{(z)} - 2g_i(u_i^{\text{best}}) \right) + \sum_{j \in I_2^{(z)}} \xi^{(z)} \left(\lambda_j^{(z)} - \bar{\lambda}^{(z)} \right) \left(\theta_j^{(z+1)} + \theta_j^{(z)} - 2g_j(u_j^{\text{best}}) \right) \\
&= \sum_{i=1}^{N} -\xi^{(z)} \bar{\lambda}^{(z)} \left(\theta_i^{(z+1)} + \theta_i^{(z)} - 2g_i(u_i^{\text{best}}) \right) + \sum_{j \in I_2^{(z)}} \xi^{(z)} \lambda_j^{(z)} \left(\theta_j^{(z+1)} + \theta_j^{(z)} - 2g_j(u_j^{\text{best}}) \right)
\end{aligned}$$

Further, since

$$\begin{aligned}
&\sum_{i=1}^{N} -\xi^{(z)} \bar{\lambda}^{(z)} \left(\theta_i^{(z+1)} + \theta_i^{(z)} - 2g_i(u_i^{\text{best}}) \right) \\
&= -\xi^{(z)} \bar{\lambda}^{(z)} \left(\sum_{i=1}^{N} \theta_i^{(z+1)} + \sum_{i=1}^{N} \theta_i^{(z)} - 2 \sum_{i=1}^{N} g_i(u_i^{\text{best}}) \right) \\
&= -2\xi^{(z)} \bar{\lambda}^{(z)} \left(r - \sum_{i=1}^{N} g_i(u_i^{\text{best}}) \right) \\
&\leq -2\xi^{(z)} \frac{\delta^{\min}}{N} \left(r - \sum_{i=1}^{N} g_i(u_i^{\text{best}}) \right)
\end{aligned}$$

and

$$\begin{aligned}
&\sum_{j \in I_2^{(z)}} \xi^{(z)} \lambda_j^{(z)} \left(\theta_j^{(z+1)} + \theta_j^{(z)} - 2g_j(u_j^{\text{best}}) \right) \\
&= \sum_{j \in I_2^{(z)}} \xi^{(z)} \lambda_j^{(z)} \left(\theta_j^{(z)} + \xi^{(z)} \left(\lambda_j^{(z)} - \bar{\lambda}^{(z)} \right) + \theta_j^{(z)} - 2g_j(u_j^{\text{best}}) \right) \\
&= \sum_{j \in I_2^{(z)}} \left(\xi^{(z)} \right)^2 \lambda_j^{(z)} \left(\lambda_j^{(z)} - \bar{\lambda}^{(z)} \right) + \sum_{j \in I_2^{(z)}} 2\xi^{(z)} \lambda_j^{(z)} \left(\theta_j^{(z)} - g_j(u_j^{\text{best}}) \right) \\
&< \sum_{j \in I_2^{(z)}} \left(\xi^{(z)} \right)^2 \lambda_j^{(z)} \left(\lambda_j^{(z)} - \bar{\lambda}^{(z)} \right) \\
&< \sum_{j \in I_2^{(z)}} \left(\xi^{(z)} \right)^2 (\sigma^{\max})^2 < N \cdot \left(\xi^{(z)} \right)^2 (\sigma^{\max})^2
\end{aligned}$$

we have

$$J(z+1) - J(z) < -2\xi^{(z)} \frac{\delta^{\min}}{N} \left(r - \sum_{i=1}^{N} g_i(u_i^{\text{best}}) \right) + N \cdot \left(\xi^{(z)} \right)^2 (\sigma^{\max})^2$$

Moreover, let K be an arbitrary finite integer. Then, if $I_2^{(z)} \neq \emptyset$ at any of step $z \in \{1, 2, ..., K\}$, we have

$$J(K+1) < J(1) - \frac{2\delta^{\min}}{N} \left(r - \sum_{i=1}^{N} g_i(u_i^{\text{best}}) \right) \sum_{z=1}^{K} \xi^{(z)} + N \cdot (\sigma^{\max})^2 \sum_{z=1}^{K} \left(\xi^{(z)} \right)^2$$

Since $\sum_{z=1}^{+\infty}\xi^{(z)}=+\infty$ and $\sum_{z=1}^{+\infty}(\xi^{(z)})^2<+\infty$, we can always select K such that

$$J(1)-\frac{2\delta^{\min}}{N}\Big(r-\sum_{i=1}^{N}g_i(u_i^{\text{best}})\Big)\sum_{z=1}^{K}\xi^{(z)}+N\cdot(\sigma^{\max})^2\sum_{z=1}^{K}\big(\xi^{(z)}\big)^2<0$$

Then we obtain

$$J(K+1)<0$$

This contradicts the fact that $J(\cdot)$ is a nonnegative function. Therefore, the assumption that $I_2^{(z)}\neq\emptyset$ at any of step $z\in\{1,2,...,K\}$ does not hold, i.e. $I_2^{(M)}=\emptyset$ at some step $M\leq K$.

Since we have proved that $I_2^{(M)}=\emptyset$ at some step $M\leq K$, then we have $\theta_i^{(M)}\geq g_i(u_i^{\text{best}})$, $u_i^{*,(M)}=u_i^{\text{best}}$ and $\lambda_i^{*,(M)}=0$ for all $i\in I$. Therefore, we have for all $i\in I$

$$\theta_i^{(z)}=\theta_i^{(M)},\ \ \forall z\geq M$$

and $\theta_i^{(z)}\geq g_i(u_i^{\text{best}})$ holds for all $z\geq M$. It is proved that $I_2^{(z)}$ is empty for all $z\geq M$. □

Proposition 5.2: In the case $\sum_{i=1}^{N}g_i(u_i^{\text{best}})<r$, there exists a finite integer $M\geq 0$ such that the overall optimal solution is attained at step $z=M$.
Proof: According to Proposition 5.1, there exists a finite integer $M\geq 0$ such that $I_2^{(M)}=\emptyset$. Then, $\theta_i^{(M)}\geq g_i(u_i^{\text{best}})$ holds for all $i\in I$ and we have $u_i^{*,(M)}=u_i^{\text{best}}$ for all $i\in I$. Since $f_i(u_i)\geq f_i(u_i^{\text{best}})$ holds for all $i\in I$, $u_i^{*,(M)}=u_i^{\text{best}}$ with $i\in I$ is the overall optimal solution. Note that since $\theta_i^{(z)}=\theta_i^{(M)}$ for all $z>M$, we have $u_i^{*,(z)}=u_i^{*,(M)}=u_i^{\text{best}}$. Therefore, the overall optimal solution is also attained at step $z>M$. □

For Case 2

Proposition 6.1: For any $\varepsilon>0$, given $\sum_{i=1}^{N}g_i(x_i^{\text{best}})=r-\varepsilon$, there exists a finite $M\geq 0$ such that at any step $z\geq M$, $I_2^{(z)}=\emptyset$.
Proof: We first assume that I_2 will never be empty and then obtain a contradiction. Next, we show that once I_2 is empty, it stays empty afterwards.

In the proof of Proposition 5.1, we have derived that at any step z, if $I_2^{(z)}\neq\emptyset$, then we have

$$J(z+1)-J(z)<-2\xi^{(z)}\frac{\delta^{\min}}{N}\Big(r-\sum_{i=1}^{N}g_i(u_i^{\text{best}})\Big)+N\cdot\big(\xi^{(z)}\big)^2(\sigma^{\max})^2$$

Since $\sum_{i=1}^{N}g_i(u_i^{\text{best}})=r-\varepsilon$, we have

$$J(z+1)-J(z)<-2\xi^{(z)}\frac{\delta^{\min}}{N}\cdot\varepsilon+N\cdot\big(\xi^{(z)}\big)^2(\sigma^{\max})^2$$

First, let K be an arbitrary finite integer. Then, if $I_2^{(z)}\neq\emptyset$ at each of step $z\in\{1,2,...,K\}$, we have

$$J(K+1)<J(1)-\frac{2\delta^{\min}}{N}\cdot\varepsilon\cdot\sum_{z=1}^{K}\xi^{(z)}+N\cdot(\sigma^{\max})^2\sum_{z=1}^{K}\big(\xi^{(z)}\big)^2$$

Since $\sum_{z=1}^{+\infty} \xi^{(z)} = +\infty$ and $\sum_{z=1}^{+\infty} (\xi^{(z)})^2 < +\infty$, for any $\varepsilon > 0$, we can always select K such that

$$J(1) - \frac{2\delta^{\min}}{N} \cdot \varepsilon \cdot \sum_{z=1}^{K} \xi^{(z)} + N \cdot (\sigma^{\max})^2 \sum_{z=1}^{K} \left(\xi^{(z)}\right)^2 < 0$$

Hence, we obtain

$$J(K+1) < 0$$

This contradicts the fact that $J(\cdot)$ is a nonnegative function. Therefore, the assumption that $I_2^{(z)} \neq \emptyset$ at each step $z \in \{1, 2, ..., K\}$ does not hold. That is to say, $I_2^{(M)} = \emptyset$ at some step $M \leq K$.

Since we have proved that $I_2^{(M)} = \emptyset$ at some step $M \leq K$, then we have $\theta_i^{(M)} \geq g_i(u_i^{\text{best}})$, $u_i^{*,(M)} = u_i^{\text{best}}$ and $\lambda_i^{*,(M)} = 0$ for all $i \in I$. Therefore, we have for all $i \in I$

$$\theta_i^{(z)} = \theta_i^{(M)}, \ \ \forall z \geq M$$

and $\theta_i^{(z)} \geq g_i(u_i^{\text{best}})$ holds for all $z \geq M$. Hence, it has been proved that $I_2^{(z)}$ is empty for all $z \geq M$. □

Proposition 6.2: For any $\varepsilon > 0$, given $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) = r - \varepsilon$, there exists a finite integer $M \geq 0$ such that at any step $z \geq M$, $0 \leq \theta_i^{(z)} - g_i(u_i^{\text{best}})$ for all $i \in I$ and $\sum_{i=1}^{N} \left(\theta_i^{(z)} - g_i(u_i^{\text{best}})\right) = \varepsilon$.
Proof:
According to Proposition 6.1, there exists a finite integer $M \geq 0$ such that at all step $z \geq M$, $I_2^{(z)} = \emptyset$. Therefore, at any step $z \geq M$, we have $\theta_i^{(z)} \geq g_i(u_i^{\text{best}})$ for all $i \in I$. Since $\theta_i^{(z)} - g_i(u_i^{\text{best}}) \geq 0$ at all step $z \geq M$, we have for all $i \in I$, $\theta_i^{(z)} - g_i(u_i^{\text{best}}) \leq \sum_{i=1}^{N} \left(\theta_i^{(z)} - g_i(u_i^{\text{best}})\right)$. Further, since $\sum_{i=1}^{N} g_i(u_i^{\text{best}}) = r - \varepsilon$ and $\sum_{i=1}^{N} \theta_i^{(z)} = r$, we have $\sum_{i=1}^{N} \left(\theta_i^{(z)} - g_i(u_i^{\text{best}})\right) = \varepsilon$ and $\theta_i^{(z)} - g_i(u_i^{\text{best}}) \leq \varepsilon$. □

Bibliography

[1] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

[2] B. Abdulhai, R. Pringle, and G. J. Karakoulas. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, 129(3):278–285, 2003.

[3] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[4] I. Alvarado, D. Limon, D. M. de la Pena, J.M. Maestre, M.A. Ridao, H. Scheu, W. Marquardt, R.R. Negenborn, B. De Schutter, F. Valencia, et al. A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark. *Journal of Process Control*, 21(5):800–815, 2011.

[5] F. An, M. Barth, J. Norbeck, and M. Ross. Development of comprehensive modal emissions model: Operating under hot-stabilized conditions. *Transportation Research Record*, (1587):52–62, 1997.

[6] K. J. Åström and B. Wittenmark. *Computer-Controlled systems: Theory and Design*. Courier Corporation, 2013.

[7] A. Awasthi, S. S. Chuanhan, M. Parent, and J. M. Proth. Centralized fleet management system for cybernetic transportation. *Expert Systems with Applications*, 38(4):3710–3717, 2011.

[8] J. Barceló. *Fundamentals of Traffic Simulation*, volume 145. Springer Science & Business Media, 2010.

[9] T. Basar and G. J. Olsder. *Dynamic Non-Cooperative Game Theory*. 2nd edition, Academic Press, 1995.

[10] L.D. Baskar, B. De Schutter, and H. Hellendoorn. Optimal routing for automated highway systems. *Transportation Research Part C*, 30:1–22, May 2013.

[11] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[12] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, 45(10): 1864–1876, 2000.

[13] T. Berger, Y. Sallez, S. Raileanu, C. Tahon, D. Trentesaux, and T. Borangiu. Personal rapid transit in an open-control framework. *Computer & Industrial Engineering*, 61 (2):300–312, 2011.

[14] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, 1995.

[15] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[16] R. Bishop. *Intelligent Vehicles Technology and Trends*. Artech House, 2005.

[17] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford university press, 1999.

[18] A. Bose and P. Ioannou. Mixed manual/semi-automated traffic: a macroscopic analysis. *Transportation Research Part C: Emerging Technologies*, 11(6):439–462, 2003.

[19] R. Bourdais, H. Guéguen, and A. Belmiloudi. Distributed model predictive control for a class of hybrid system based on Lagrangian relaxation. In *Proceedings of the 4th Conference on Analysis and Design of Hybrid Systems*, pages 46–51, Eindhoven, The Netherlands, 2012.

[20] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[21] X. Cai, L. Xie, P. Lu, and J. Chen. Optimal selection of the decomposition structure based on GA for distributed model predictive control systems. In *Proceedings of 11th World Congress on Intelligent Control and Automation*, pages 4560–4565, Shenyang, China, June 2014.

[22] E. F. Camacho and C. Bordons. *Model Predictive Control in Process Industry*. Springer-Verlag, Berlin, Germany, 1995.

[23] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1):44–52, 2002.

[24] Y. Cao, W. Yu, W. Ren, and G. Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9 (1):427–438, 2013.

[25] A. Cardillo, J. Gómez-Gardeñes, M. Zanin, M. Romance, D. Papo, F. del Pozo, and S. Boccaletti. Emergence of network features from multiplexity. *Scientific Reports*, 3 (1344), 2013.

[26] M. Carey and E. Subrahmanian. An approach to modeling time-varying flows on congested networks. *Transportation Research Part B: Methodological*, 34(3):157–183, 2000.

[27] P. D. Christofides, R. Scattolini, D. M. de la Peña, and J. Liu. Distributed model predictive control: A tutorial review and future research directions. *Computers & Chemical Engineering*, 51:21–41, 2013.

[28] K. Clement-Nyns, E. Haesen, and J. Driesen. The impact of charging plug-in hybrid electric vehicles on a residential distribution grid. *IEEE Transactions on Power Systems*, 25(1):371–380, 2010.

[29] G. Cohen. Optimization by decomposition and coordination: a unified approach. *IEEE Transactions on Automatic Control*, 23(2):222–232, 1978.

[30] Z. Cong, B. De Schutter, and R. Babuška. Ant colony routing algorithm for freeway networks. *Transportation Research Part C: Emerging Technologies*, 37:1–19, 2013.

[31] D. Corona and B. De Schutter. Adaptive cruise control for a SMART car: A comparison benchmark for MPC-PWA control methods. *IEEE Transactions on Control Systems Technology*, 16(2):365–372, 2008.

[32] C. F. Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287, 1994.

[33] C. F. Daganzo. The cell transmission model, Part II: Network traffic. *Transportation Research Part B: Methodological*, 29(2):79–93, 1995.

[34] C. F. Daganzo. *Fundamentals of Transportation and Traffic Operations*, volume 30. Pergamon Press, 1997.

[35] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1998.

[36] B. De Schutter. Optimal control of a class of linear hybrid systems with saturation. *SIAM Journal on Control and Optimization*, 39(3):835–851, 2000.

[37] B. De Schutter and B. De Moor. Optimal traffic light control for a single intersection. *European Journal of Control*, 4(3):260–276, 1998.

[38] B. De Schutter and T. Van den Boom. Model predictive control for max-min-plus systems. *Discrete Event Systems: Analysis and Control*, 569:201–208, 2000.

[39] R. Dekker, J. Bloemhof, and I. Mallidis. Operations research for green logistics–An overview of aspects, issues, contributions and challenges. *European Journal of Operational Research*, 219(3):671–679, 2012.

[40] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[41] D. Driankov, H. Hellendoorn, and M. Reinfrank. *An Introduction to Fuzzy Control*. Springer Science & Business Media, 2013.

[42] W. B. Dunbar and R. M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):549–558, 2006.

[43] P. S. Dutta, N. R. Jennings, and L. Moreau. Cooperative information sharing to improve distributed learning in multi-agent systems. *Journal of Artificial Intelligence Research*, 24:407–463, 2005.

[44] D. Eppstein. Finding the K shortest paths. *SIAM Journal on Computing*, 28(2):652–673, 1998.

[45] W. R. Esposito and C. A. Floudas. Deterministic global optimization in nonlinear optimal control problems. *Journal of Global Optimization*, 17(1-4):97–126, 2000.

[46] R. Z. Farahani, E. Miandoabchi, W. Y. Szeto, and H. Rashidi. A review of urban transportation network design problems. *European Journal of Operational Research*, 229(2):281–302, 2013.

[47] C. A. Farrell, D. H. Kieronska, and M. Schulze. Genetic algorithms for network division problem. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation*, pages 422–427, 1994.

[48] J. R. D. Frejo and E. F. Camacho. Global versus local MPC algorithms in freeway traffic control with ramp metering and variable speed limits. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1556–1565, 2012.

[49] D. Frick, A. Domahidi, and M. Morari. Embedded optimization for mixed logical dynamical systems. *Computers & Chemical Engineering*, 72:21–33, 2015.

[50] T. Garaix, C. Artigues, D. Feillet, and D. Josselin. Vehicle routing problems with alternative paths: An application to on-demand transportation. *European Journal of Operational Research*, 204(1):62–75, 2010.

[51] N. Geroliminis and C. F. Daganzo. Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings. *Transportation Research Part B: Methodological*, 42(9):759–770, 2008.

[52] A. Hajimiragha, C. A. Canizares, M. W. Fowler, and A. Elkamel. Optimal transition to plug-in hybrid electric vehicles in ontario, canada, considering the electricity-grid limitations. *IEEE Transactions on Industrial Electronics*, 57(2):690–701, 2010.

[53] S. Han, S. Han, and K. Sezaki. Development of an optimal vehicle-to-grid aggregator for frequency regulation. *IEEE Transactions on Smart Grid*, 1(1):65–72, 2010.

[54] C. Hatipoglu, U. Ozguner, and K. Redmill. Automated lane change controller design. *IEEE Transactions on Intelligent Transportation Systems*, 4(1):13–22, 2003.

[55] W. P. M. H. Heemels, J. M. Schumacher, and S. Weiland. Linear complementarity systems. *SIAM Journal on Applied Mathematics*, 60(4):1234–1269, 2000.

[56] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.

[57] A. Hegyi, B. De Schutter, and J. Hellendoorn. Optimal coordination of variable speed limits to suppress shock waves. *IEEE Transactions on Intelligent Transportation Systems*, 6(1):102–112, 2005.

[58] J. Hu, A. Saleem, S. You, L. Nordström, M. Lind, and J. Østergaard. A multi-agent system for distribution grid congestion management with electric vehicles. *Engineering Applications of Artificial Intelligence*, 38:45–58, 2015.

[59] C. Huang, C. Guan, H. Chen, Y. Wang, S. Chang, C. Li, and C. Weng. An adaptive resource management scheme in cloud computing. *Engineering Applications of Artificial Intelligence*, 26(1):382–389, 2013.

[60] J. R. Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23(3):665–685, 1993.

[61] F. V. Jensen. *An Introduction to Bayesian Networks*, volume 210. UCL press London, 1996.

[62] M. Johansson and A. Rantzer. Computation of piecewise quadratic Lyapunov functions for hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):555–559, 1998.

[63] D. R. Jones. DIRECT global optimization algorithm. In *Encyclopedia of Optimization*, pages 431–440. Springer, 2001.

[64] A. Kantamneni, L. E. Brown, G. Parker, and W. W. Weaver. Survey of multi-agent systems for microgrid control. *Engineering Applications of Artificial Intelligence*, 45: 192–203, 2015.

[65] T. Keviczky, F. Borrelli, and G. J. Balas. Decentralized receding horizon control for large scale dynamically decoupled systems. *Automatica*, 42(12):2105–2115, 2006.

[66] M. Keyvan-Ekbatani, A. Kouvelas, I. Papamichail, and M. Papageorgiou. Exploiting the fundamental diagram of urban networks for feedback-based gating. *Transportation Research Part B: Methodological*, 46(10):1393–1403, 2012.

[67] D. E. Kirk. *Optimal Control Theory: An introduction*. Dover Publications, 2004.

[68] A. Kotsialos, M. Papageorgiou, M. Mangeas, and H. Haj-Salem. Coordinated and integrated control of motorway networks via non-linear optimal control. *Transportation Research Part C: Emerging Technologies*, 10(1):65–84, 2002.

[69] A. Kotsialos, I. Papamichail, I. Margonis, and M. Papageorgiou. Hierarchical nonlinear model-predictive ramp metering control for freeway networks. In *Proceedings of the 11th IFAC Symposium on Control in Transportation Systems*, pages 124–129, Delft, The Netherlands, 2006.

[70] E. L. Lawler and D. E. Wood. Branch-and-bound methods: A survey. *Operations Research*, 14(4):699–719, 1966.

[71] P. Leitão. Agent-based distributed manufacturing control: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence*, 22(7):979–991, 2009.

[72] H. Levy and D. W. Low. A contraction algorithm for finding small cycle cutsets. *Journal of Algorithms*, 9(4):470–493, 1988.

[73] D. Liberzon. *Switching in Systems and Control*. Springer Science & Business Media, 2012.

[74] N. E. Ligterink, R. De Lange, and E. Schoen. Refined vehicle and driving-behaviour dependencies in the VERSIT+ emission model. In *Proceedings of the ETTAP Symposium*, pages 177–186, Toulouse, France, 2009.

[75] C. Lin and C. G. Lee. Neural-network-based fuzzy logic control and decision system. *IEEE Transactions on Computers*, 40(12):1320–1336, 1991.

[76] R. Luo, T. J. J. van den Boom, and B. De Schutter. Modeling of the dynamics and the energy consumption of a fleet of cybercars. In *Proceedings of the 2014 European Control Conference*, pages 720–725, Strasbourg, France, 2014.

[77] R. Luo, R. Bourdais, T. J. J. van den Boom, and B. De Schutter. Integration of resource allocation coordination and branch-and-bound. In *Proceedings of the 54th IEEE Conference on Decision and Control*, pages 4272–4277, Osaka, Japan, December 2015.

[78] R. Luo, R. Bourdais, T. J. J. van den Boom, and B. De Schutter. Properties of applying a resource allocation coordination algorithm to optimization problems with discrete decision variables. Technical Report 15-025, Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands, September 2015.

[79] R. Luo, R. Bourdais, T. J. J. van den Boom, and B. De Schutter. Multi-agent model predictive control based on resource allocation coordination for a class of hybrid systems with limited information sharing. *Submitted (minor revision) to Engineering Applications of Artificial Intelligence*, 2016.

[80] R. Luo, T. J. J. van den Boom, and B. De Schutter. Efficient routing of traffic flows for urban transportation networks. *Submitted to European Journal of Operational Research*, 2016.

[81] R. Luo, T. J. J. van den Boom, and B. De Schutter. Multi-agent dynamic routing of a fleet of cybercars. *Submitted (minor revision) to IEEE Transactions on Intelligent Transportation Systems*, 2016.

[82] R. Luo, T. J. J. van den Boom, and B. De Schutter. Parameterized dynamic routing of a fleet of cybercars. In *Proceedings of the 14th IFAC Symposium on Control in Transportation Systems*, Istanbul, Turkey, 2016.

[83] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice-Hall, Harlow, England, 2002.

[84] D. MacKay. *Sustainable Energy-Without the Hot Air*. UIT Cambridge, 2008.

[85] J. M. Maestre and R. R. Negenborn, editors. *Distributed Model Predictive Control Made Easy*. Springer, 2014.

[86] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed model predictive control based on a cooperative game. *Optimal Control Applications and Methods*, 32 (5):153–176, 2011.

[87] H. Mahmassani and S. Peeta. Network performance under system optimal and user equilibrium dynamic assignments: Implications for advanced traveler information systems. *Transportation Research Record*, 1408(3):83–93, 1993.

[88] H. Mahmassani and S. Peeta. System optimal dynamic assignment for electronic route guidance in a congested traffic network. *Urban Traffic Networks: Dynamic Flow Modeling and Control*, 1408:2–27, 1995.

[89] H. Mahmassani, T. Hu, and R. Jayakrishnan. Dynamic traffic assignment and simulation for advanced network informatics DYNASMART. In *Proceedings of the 2nd international CAPRI seminar on Urban Traffic Networks, Capri, Italy*, 1992.

[90] F. Marra, G. Yang, C. Traholt, E. Larsen, C. Rasmussen, and S. You. Demand profile study of battery electric vehicle under different charging options. In *Proceedings of the 2012 IEEE Power and Energy Society General Meeting*, pages 1–7, San Diego, USA, 2012.

[91] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[92] A. Messner and M. Papageorgiou. Metanet: A macroscopic simulation program for motorway networks. *Traffic Engineering and Control*, 31(8-9):466–470, 1990.

[93] F Middelham. State of practice in dynamic traffic management in the netherlands. In *Proceedings of the 10th IFAC Symposium on Control in Transportation Systems*, Tokyo, Japan, 2003.

[94] P. D. Moroşan, R. Bourdais, D. Dumur, and J. Buisson. A distributed MPC strategy based on benders' decomposition applied to multi-source multi-zone temperature regulation. *Journal of Process Control*, 21(5):729–737, 2011.

[95] R. R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. *Engineering Applications of Artificial Intelligence*, 21(3):353–366, 2008.

[96] S. Olaru, J. Thomas, D. Dumur, and J. Buisson. Genetic algorithm based model predictive control for hybrid systems under a modified MLD form. *International Journal of Hybrid Systems*, 4(1-2):113–132, 2004.

[97] R. Oung, M. P. Cruz, and R. D'Andrea. A parameterized control methodology for a modular flying vehicle. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 532–538, Algarve, Portugal, 2012.

[98] D. P. Palomar and M. Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8):1439–1451, 2006.

[99] M. Papageorgiou. *Applications of Automatic Control Concepts to Traffic Flow Modeling and Control*. Lecture Notes in Control and Information Sciences. Springer-Verlag, 1983.

[100] P. M. Pardalos and M. G. C. Resende, editors. *Handbook of Applied Optimization*. Oxford University Press, Oxford, UK, 2002.

[101] C. K. Park, D. J. Scheeres, V. T. Guibout, and A. Bloch. Global solution for the optimal feedback control of the underactuated heisenberg system. *IEEE Transactions on Automatic Control*, 53(11):2638–2642, 2008.

[102] A. Paz and S. Peeta. Behavior-consistent real-time traffic routing under information provision. *Transportation Research Part C: Emerging Technologies*, 17(6):642–661, 2009.

[103] A. Paz and S. Peeta. Information-based network control strategies consistent with estimated driver behavior. *Transportation Research Part B: Methodological*, 43(1):73–96, 2009.

[104] S. Polinder, J. Haagsma, N. Bos, M. Panneman, K. K. Wolt, M. Brugmans, W. Weijermars, and E. van Beeck. Burden of road traffic injuries: Disability-adjusted life years in relation to hospitalization and the maximum abbreviated injury scale. *Accident Analysis & Prevention*, 80:193–200, 2015.

[105] S. V. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen. Parameterized tube model predictive control. *IEEE Transactions on Automatic Control*, 57(11):2746–2761, 2012.

[106] L. Rambaldi, E. Bocci, and F. Orecchini. Preliminary experimental evaluation of a four wheel motors, batteries plus ultracapacitors and series hybrid powertrain. *Applied Energy*, 88(2):442–448, 2011.

[107] B. Ran, D. E. Boyce, and L. J. LeBlanc. A new class of instantaneous dynamic user-optimal traffic assignment models. *Operations Research Record*, 41(1):192–202, 1993.

[108] J. B. Rawlings and B. T. Stewart. Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9):839–845, 2008.

[109] P. I. Richards. Shock waves on the highway. *Operations Research*, 4(1):42–51, 1956.

[110] P. Rode and G. Floater. Accessibility in cities: transport and urban form. *New Climate Economy Cities*, (3):1–60, 2014.

[111] A. Y. Saber and G. K. Venayagamoorthy. Plug-in vehicles and renewable energy sources for cost and emission reductions. *IEEE Transactions on Industrial Electronics*, 58(4): 1229–1238, 2011.

[112] R. Scattolini. Architectures for distributed and hierarchical model predictive control–A review. *Journal of Process Control*, 19(5):723–731, 2009.

[113] R. Scattolini and P. Colaneri. Hierarchical model predictive control. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 4803–4808, New Orleans, USA, 2007.

[114] P. Shi, Y. Xia, G. P. Liu, and D. Rees. On designing of sliding-mode control for stochastic jump systems. *IEEE Transactions on Automatic Control*, 51(1):97–103, 2006.

[115] N. Z. Shor. *Minimization Methods for Non-Differentiable Functions*. Springer Science & Business Media, 2012.

[116] D. D. Siljak. *Decentralized Control of Complex Systems*. Courier Corporation, 2011.

[117] E. D. Sontag. Nonlinear regulation: the piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346–358, 1981.

[118] M. Sørensen, P. Lühdorf, M. Ketzel, Z. J. Andersen, A. Tjønneland, K. Overvad, and O. Raaschou-Nielsen. Combined effects of road traffic noise and ambient air pollution in relation to risk for stroke. *Environmental Research*, 133:49–55, 2014.

[119] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia. Cooperative distributed model predictive control. *Systems & Control Letters*, 59(8):460–469, 2010.

[120] S. Suri, M. Waldvogel, D. Bauer, and P. R. Warkhede. Profile-based routing and traffic engineering. *Computer Communications*, 26(4):351–365, 2003.

[121] H. J. Sussmann and J. C. Willems. 300 years of optimal control: From the brachystochrone to the maximum principle. *IEEE Control Systems Magazine*, 17(3): 32–44, 1997.

[122] J. L. Szwarcfiter and P. E. Lauer. A search strategy for the elementary cycles of a directed graph. *BIT Numerical Mathematics*, 16(2):192–204, 1976.

[123] A. N. Tarău, B. De Schutter, and J. Hellendoorn. Model-based control for route choice in automated baggage handling systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 40(3):341–351, 2010.

[124] C. J. Tomlin, J. Lygeros, and S. S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, 2000.

[125] M. Treiber and A. Kesting. *Traffic Flow Dynamics: Data, Models and Simulation*. Springer-Verlag, 2010.

[126] T. Van Woensel, L. Kerbache, H. Peremans, and N. Vandaele. Vehicle routing with dynamic travel times: A queueing approach. *European Journal of Operational Research*, 186(3):990–1007, 2008.

[127] P. Vas. *Artificial-Intelligence-Based Electrical Machines and Drives: Application of Fuzzy, Neural, Fuzzy-Neural, and Genetic-Algorithm-Based Techniques*, volume 45. Oxford University Press, 1999.

[128] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright. Distributed MPC strategies with application to power system automatic generation control. *IEEE Transactions on Control Systems Technology*, 16(6):1192–1206, 2008.

[129] L. Vlacic, M. Parent, and F. Harashima. *Intelligent Vehicle Technologies: Theory and Applications*. Butterworth-Heinemann, 2001.

[130] R. Vujanic, P. M. Esfahani, P. Goulart, S. Mariethoz, and M. Morari. A decomposition method for large scale MILPs, with performance guarantees and a power system application. *Automatica*, 67:144–156, 2016.

[131] L. Wu and D. Ho. Sliding mode control of singular stochastic hybrid systems. *Automatica*, 46(4):779–783, 2010.

[132] S. Yagar and M. Van Aerde. Geometric and environmental effects on speeds of 2-lane highways. *Transportation Research Part A: General*, 17(4):315–325, 1983.

[133] L. Yu, Y. Xu, G. Song, Y. Hao, S. Guo, and Q. Shi. Development and application of macroscopic emission model for china. *Transportation Research Record*, (2123):66–75, 2009.

[134] T. Zachariadis and Z. Samaras. An integrated modeling system for the estimation of motor vehicle emissions. *Journal of the Air and Waste Management Association*, 49(9): 1010–1026, 1999.

[135] S. K. Zegeye, B. De Schutter, J. Hellendoorn, E. A. Breunesse, and A. Hegyi. A predictive traffic controller for sustainable mobility using parameterized control policies. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1420–1429, 2012.

[136] X. Zhou, X. Qin, and H. Mahmassani. Dynamic origin-destination demand estimation with multiday link traffic counts for planning applications. *Transportation Research Record: Journal of the Transportation Research Board*, 1831(1831):30–38, 2003.

Summary

Multi-Agent Control of Urban Transportation Networks and of Hybrid Systems With Limited Information Sharing

This thesis is divided into two parts. In the first part, we address the dynamic traffic routing problem for urban transportation networks. In the second part, we address multi-agent model predictive control of a class of hybrid systems with limited information sharing.

Dynamic traffic routing for urban transportation networks

In this part of the thesis, we address the dynamic traffic routing problem at different levels, namely dynamic traffic routing considering dynamics of individual vehicles, dynamic traffic routing considering dynamics of traffic flows, and co-optimization of the orientation of road sections and the routes of traffic flows. Usually, solving a dynamic traffic routing problems for a large-scale urban transportation network requires extremely high computational efforts. The main contribution of this part of the thesis consists in developing efficient solution methods for dynamic traffic routing problems with a well-balanced trade-off between the quality of solutions and the computation costs.

Thanks to the fast development of automated driving technologies, the application of cybercars, which are fully automatic road vehicles providing on-demand and door-to-door transportation service, has been pushed forward greatly. However, the lack of efficient control methods for a large number of cybercars throughout a road network is still one of the biggest challenges that hinder the widespread application of cybercars. To contribute to development of the cybernetic transportation system, we address the dynamic routing problem for cybercars considering minimization of the combined system cost including the total time spent and the total energy consumption of all cybercars. We first propose a model of the dynamics and the energy consumption of cybercars based on a description of the dynamics of every single cybercar and the states, e.g. traffic densities, of the road network. After that, we propose several tractable and scalable multi-agent control methods including multi-agent model predictive control and parameterized control for dynamic routing of cybercars.

Next, we propose two novel multi-agent dynamic traffic routing methods considering dynamics of traffic flows, namely hierarchical traffic routing based on network division and bi-level traffic routing based on merging nodes and links. In the hierarchical traffic routing approach, a large-scale network is divided into a group of subnetworks and each subnetwork is assigned a local subnetwork traffic routing controller. At the higher level, the interconnecting flows among subnetworks are determined by a centralized network traffic

routing controller. After that, the flows within each subnetwork are determined by the corresponding subnetwork traffic routing controller considering minimization of the total travel cost in the subnetwork and minimization of the difference between the total traffic flows on the boundaries of the subnetwork and the flows prescribed by the high-level controller. In the bi-level traffic routing approach, nodes and links of a network are merged into a set of aggregated nodes and a set of aggregated links. For each aggregated node, a local traffic routing controller is assigned. At the aggregated level, the flows on the aggregated links are determined by a centralized traffic routing controller. At the original level, where the actual network is considered, the flows on the actual links are determined by the local traffic routing controllers by negotiating with neighboring local controllers.

Finally, we address the co-optimization of the orientation of road sections and the routes of traffic flows. Conventionally, after an urban transportation networks has been constructed, the orientations of road sections are fixed. As people commute between home and work daily, the roads directed from the city center to outside the city center are often not sufficiently used in the morning rush hours. A similar argument holds for the roads directed from outside the city center to the city center in the evening rush hours. To solve this problem, we assume that the orientation of road sections can be changed at each control period and we formulate the co-optimization problem of jointly determining the orientation of road sections and the routes of traffic flows as a mixed-integer linear programming problem. By assuming circular orientation of traffic flows in each elementary cycle in the graph corresponding to the network, we explicitly model the orientations of road sections using the orientations of these elementary cycles. Since the number of elementary cycles is much smaller than the number of road sections, the number of binary variables involved in the co-optimization problem is reduced substantially w.r.t considering independent orientation of each road section. Therefore, the resulting co-optimization problem can be solved more efficiently.

Control of a class of hybrid systems with limited information sharing

Control of large-scale systems, like transportation systems, manufacturing systems, power systems, etc., is facing a variety of challenges. Among the challenges, a crucial one is to design mechanisms for coordinating agents that have limited information sharing with each other in order to protect confidential information of local subsystems while at the same time still aiming for global performance. Thus, to achieve globally satisfactory performance, given limited information of other subsystems, the agents need to assist each other to make better decisions about their actions. However, cooperation among agents is made much more difficult when the individual agents have to regulate subsystems that are governed by discrete inputs instead of by continuous inputs. In fact, this will result in having to solve integer programming problems in a distributed way.

In this part of the thesis, we develop a multi-agent model predictive control method for a class of hybrid systems governed by discrete inputs and subject to global hard constraints. We assume that for each subsystem the local objective function is convex and the local constraint function is strictly increasing with respect to the local control variable. The proposed control method is based on a distributed resource allocation coordination algorithm and only requires limited information sharing among the local agents of the subsystems. Thanks to primal decomposition of the global constraints, the distributed

algorithm can always guarantee global feasibility of the local control decisions, even in the case of premature termination. However, since the control variables are discrete, global optimality cannot be attained by only using the resource allocation coordination algorithm. In order to tackle this problem, a mechanism is developed to branch the overall solution space based on the outcome of the resource allocation coordination algorithm at each node of the search tree. Finally, results for the charging control of a fleet of electric vehicles under constrained grid condition in a simulation study show that the proposed control method effectively balances the solution quality, the computation time, and the communication burden.

Renshi Luo

Samenvatting

Multi-Agent Regelen van Stedelijke Vervoersnetwerken en Hybride Systemen met Beperkte Uitwisseling van Informatie

Dit proefschrift bestaat uit twee delen: Het eerste deel gaat over dynamische verkeersroutering in stedelijke vervoersnetwerken. Het tweede deel behandeld modelgebaseerd voorspellend regelen voor een klasse van hybride systemen met beperkte uitwisseling van informatie.

Dynamische verkeersroutering voor stedelijke vervoersnetwerken

In dit deel van de thesis behandelen we het dynamische verkeersrouteringsprobleem op verschillende niveaus, namelijk op het niveau van de individuele voertuigen en op het niveau van de verkeersstromen. Verder beschouwen we de gelijktijdige optimalisatie van de rijrichting van wegdelen en de routes van verkeersstromen. Normaal gesproken vereist het oplossen van een dynamisch verkeersrouteringsprobleem voor een groot stedelijk vervoersnetwerk enorm veel rekenkracht. De hoofdbijdrage van dit onderdeel van dit proefschrift ligt dan ook in de ontwikkeling van efficiënte oplossingsstrategieën voor dynamische verkeersrouteringsproblemen waarbij we een evenwichtige afweging maken tussen de kwaliteit van oplossingen en de rekenkosten.

Dankzij de snelle ontwikkeling van geautomatiseerde rijtechnologieën is de toepassing van cybercars, zijnde volledig geautomatiseerde voertuigen die op aanvraag een deur-tot-deur service bieden, sterk naar voren geschoven. De verdere uitbreiding van de toepassing van cybercars wordt echter nog belemmerd door de afwezigheid van efficiënte regelstrategieën voor grote aantallen cybercars in een wegennet. Om bij te dragen aan de verdere ontwikkeling en toepassing van cybercars behandelen we het dynamische routeringsprobleem voor cybercars waarbij we ons richten op het minimaliseren van de totale kosten, welke onder andere afhangen van de rijtijd en energieconsumptie van alle voertuigen in het netwerk. Als eerste presenteren we een model van de dynamica en energieconsumptie van cybercars. Dit model is gebaseerd op de dynamica van de afzonderlijke voertuigen en op de toestand van het netwerk, met inbegrip van onder andere de verkeersdrukte. Aansluitend presenteren we verschillende handel- en schaalbare multi-agent regelstrategieën, waaronder multi-agent modelgebaseerd voorspellend regelen en geparametriseerd regelen, voor de dynamische routering van cybercars.

Vervolgens presenteren we twee nieuwe multi-agent dynamische verkeersrouteringsmethoden die rekening houden met de dynamica van verkeersstromen: hiërarchische verkeersroutering op basis van netwerkdivisie, en verkeersroutering op twee

niveaus gebaseerd op het samenvoegen van knooppunten en links. In de hiërarchische methode voor verkeersrouting wordt het netwerk verdeeld in verschillende deelnetwerken. Aan elk van de deelnetwerken wordt een lokale verkeersrouteringsregelaar toegewezen. Op centraal niveau worden de verkeersstromen tussen de onderling verbonden deelnetwerken bepaald door een centrale verkeersregelaar. Vervolgens worden de verkeersstromen in de deelnetwerken bepaald door de bijbehorende lokale regelaar waarbij we de totale reiskosten in het deelnetwerk minimaliseren en tegelijkertijd de verkeersstromen tussen de verschillende deelnetwerken zo goed mogelijk laten overeenkomen met de verkeersstromen bepaald door de overkoepelende regelaar. De bi-level verkeersrouteringsregelaar voegt verschillende knooppunten en links samen. Aan elk van de samengestelde knooppunten wordt een lokale regelaar toegewezen. De verkeersstromen op de samengestelde links worden bepaald door een overkoepelende regelaar. De verkeersstromen op de oorspronkelijke links worden vervolgens bepaald door de lokale regelaars die onderhandelen met de regelaars van naastgelegen links.

Tenslotte adresseren we de gelijktijdige optimalisatie van de rijrichting van wegdelen en de routes van verkeersstromen. Gewoonlijk ligt de rijrichting van een wegdeel vast na constructie. Aangezien mensen pendelen tussen huis en werk worden de wegdelen die het verkeer de stad uitleiden niet volledig benut in de ochtendspits, en de wegdelen die het verkeer de stad inleiden niet volledig benut in de avondspits. Om dit probleem aan te pakken, veronderstellen we dat de rijrichting van elk wegdeel op elk regeltijdstip aangepast kan worden, en formuleren we het probleem van de gelijktijdige optimalisatie van de rijrichting van wegdelen en de routes van verkeersstromen als een lineair programmeerprobleem met variabelen die reële dan wel gehele getalswaarden aannemen. Door het veronderstellen van circulaire verkeersstroomrichtingen in elke elementaire cirkel van de netwerkgraaf, modelleren we expliciet de richting van de wegdelen gebruikmakend van de richtingen van deze elementaire cirkels. Aangezien het aantal elementaire cirkels veel kleiner is dan het aantal wegdelen, is het aantal te optimaliseren binaire variabelen aanzienlijk verminderd ten opzichte van het afzonderlijk beschouwen van elk wegdeel. Het resulterend optimalisatieprobleem kan hierdoor efficiënter worden opgelost.

Het regelen van een klasse hybride systemen met beperkte uitwisseling van informatie

Grootschalige systemen, zoals transport-, productie- en aandrijfsystemen, kampen met een groot aantal uitdagingen op besturingsgebied. Een cruciale uitdaging ligt in het ontwerpen van mechanismen voor coördinerende agenten die slechts beperkt informatie met elkaar kunnen uitwisselen. Door slechts beperkt informatie uit te wisselen kan vertrouwelijke informatie wordt beschermd en tegelijkertijd een zo groot mogelijke algehele prestatie worden nagestreefd. Om bevredigende algehele prestaties te behalen, moeten de verschillende agenten elkaar helpen om adequate beslissingen te maken betreffende hun acties. Zulke samenwerking is echter erg lastig wanneer de individuele agenten deelsystemen regelen die worden beheerst door discrete inputvariabelen in plaats van continue inputvariabelen. Dit resulteert erin dat een programmeerprobleem met discrete variabelen op een gedistribueerde manier moet worden opgelost. In dit deel van het proefschrift ontwikkelen we een multi-agent, modelgebaseerde voorspellende regelstrategie voor een klasse van hybride systemen die gekenmerkt worden door discrete

inputvariabelen en die onderhevig zijn aan globale, harde beperkingen. We veronderstellen dat voor elk deelsysteem de lokale doelfunctie convex is, en dat de lokale beperkingsfunctie strikt stijgend is met betrekking tot de lokale regelvariabele. De voorgestelde regelstrategie is gebaseerd op een gedistribueerd brontoewijzingsbeleid, en vereist enkel beperkte uitwisseling van informatie tussen de lokale agenten van de verschillende deelsystemen. Dankzij de primaire ontleding van de globale beperkingen kan het gedistribueerde algoritme altijd de globale haalbaarheid van de lokale regelbesluiten garanderen, ook in het geval van vroegtijdige beëindiging. Echter, aangezien we te maken hebben met discrete inputvariabelen, kan het globale optimum niet worden bereikt door enkel gebruik te maken van het brontoewijzingsbeleid. Om dit probleem aan te pakken, hebben we een mechanisme ontwikkeld om de totale oplossingsruimte te vertakken op basis van de uitkomst van het brontoewijzingsbeleid bij elk knooppunt van de zoekboom. Een simulatiestudie betreffende het opladen van een wagenpark in de aanwezigheid van een beperkt elektriciteitsnet laat zien dat de voorgestelde regelstrategie een goede afweging maakt tussen de kwaliteit van oplossingen, de rekentijd en de communicatiebelasting.

Renshi Luo

Curriculum Vitae

Renshi Luo was born on 30th of August 1986, in Guangxi, China. He received B.Sc and M.Sc degree in Control Engineering from Beijing Jiaotong University, Beijing, China, in 2009 and 2012, respectively. His master thesis "Study of fault-tolerant control synthesis of urban rail vehicle" was carried out under the supervision of Dr. Zhenyu Yu and Prof. Tao Tang, and was awarded the outstanding master thesis award by Beijing Jiaotong University in 2012. Renshi Luo also worked as a research trainee in IBM Research China from August to December in 2011 with research topic "Development of intelligent access control methods for sever overload protection". Together with his supervisor Haishan Wu and colleague Asser N. Tantawi, they have applied a patent (US9124505) "System detection method and apparatus and flow control method and device" for their joint work.

Since September 2012, Renshi Luo has been working toward the Ph.D degree in Systems and Control at the Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands. The research of his Ph.D project focused on two main topics, namely urban traffic routing and multi-agent model predictive control of hybrid systems. This research is supported by the Chinese Scholarship Council and has been performed under the supervision of Prof.dr.ir Bart De Schutter and Dr.ir. Ton J.J. van den Boom. During his Ph.D research, he cooperated with Prof. Romain Bourdais from CentraleSupelec, France.

Renshi Luo has obtained the DISC certificate for the course program of the Dutch Institute for Systems and Control. Since 2012, Renshi Luo has been a member of the Dutch research school for Transport, Infrastructure and Logistics (TRAIL).

The research interests of Renshi Luo include multi-agent control, distributed and large-scale systems, and intelligent transportation and infrastructure systems.

TRAIL Thesis Series

The following list contains the most recent dissertations in the TRAIL Thesis Series. For a complete overview of more than 100 titles see the TRAIL website: www.rsTRAIL.nl.

The TRAIL Thesis Series is a series of the Netherlands TRAIL Research School on transport, infrastructure and logistics.

Luo, R., *Multi-Agent Control of Urban Transportation Networks and of Hybrid Systems With Limited Information Sharing*, T2016/21, November 2016, TRAIL Thesis Series, the Netherlands

Campanella, M., *Microscopic Modelling of Walking Behavior*, T2016/20, November 2016, TRAIL Thesis Series, the Netherlands

Horst, M. van der, *Coordination in Hinterland Chains: An Institutional Analysis of Port-related Transport*, T2016/19, November 2016, TRAIL Thesis Series, the Netherlands

Beukenkamp, *Securing Safety: Resilience Time as A Hidden Critical Factor*, T2016/18, October 2016, TRAIL Thesis Series, the Netherlands

Mingardo, G., *Articles on Parking Policy*, T2016/17, October 2016, TRAIL Thesis Series, the Netherlands

Duives, D.C., *Analysis and Modelling of Pedestrian Movement Dynamics at Large-Scale Events*, T2016/16, October 2016, TRAIL Thesis Series, the Netherlands

Wan Ahmad, W.N.K., *Contextual Factors of Sustainable Supply Chain Management Practices in the Oil and Gas Industry*, T2016/15, September 2016, TRAIL Thesis Series, the Netherlands

Liu, X., *Prediction of Belt Conveyor Idler Performance*, T2016/14, September 2016, TRAIL Thesis Series, the Netherlands

Gaast, J.P. van der, *Stochastic Models for Order Picking Systems*, T2016/13, September 2016, TRAIL Thesis Series, the Netherlands

Wagenaar, J.C., *Practice Oriented Algorithmic Disruption Management in Passenger Railways*, T2016/12, September 2016, TRAIL Thesis Series, the Netherlands

Psarra, I., *A Bounded Rationality Model of Short and Long-Term Dynamics of Activity-Travel Behavior*, T2016/11, June 2016, TRAIL Thesis Series, the Netherlands

Ma, Y., *The Use of Advanced Transportation Monitoring Data for Official Statistics*, T2016/10,

June 2016, TRAIL Thesis Series, the Netherlands

Li, L., *Coordinated Model Predictive Control of Synchromodal Freight Transport Systems*, T2016 /9, June 2016, TRAIL Thesis Series, the Netherlands

Vonk Noordegraaf, D.M., *Road Pricing Policy Implementation*, T2016/8, June 2016, TRAIL Thesis Series, the Netherlands

Liu, S., *Modeling, Robust and Distributed Model Predictive Control for Freeway Networks*, T2016/7, May 2016, TRAIL Thesis Series, the Netherlands

Calvert, S., *Stocahstic Macroscopic Analysis and Modelling for Traffic Management*, T2016/6, May 2016, TRAIL Thesis Series, the Netherlands

Sparing, D., *Reliable Timetable Design for Railways and Connecting Public Transport Services*, T2016/5, May 2016, TRAIL Thesis Series, the Netherlands

Rasouli, S., *Uncertainty in Modeling Activity-Travel Demand in Complex Urban Systems*, T2016 /4, March 2016, TRAIL Thesis Series, the Netherlands

De Vries, J., *Behavioral Operations in Logistics*, T2016/3, February 2016, TRAIL Thesis Series, the Netherlands

Goni-Ros, B., *Traffic Flow at Sags: Theory, Modeling and Control*, T2016/2, February 2016, TRAIL Thesis Series, the Netherlands

Khademi, E., *Effects of Pricing Strategies on Dynamic Repertoires of Activity-Travel Behaviour*, T2016/1, February 2016, TRAIL Thesis Series, the Netherlands

Cong, Z., *Efficient Optimization Methods for Freeway Management and Control*, T2015/17, November 2015, TRAIL Thesis Series, the Netherlands

Kersbergen, B., *Modeling and Control of Switching Max-Plus-Linear Systems: Rescheduling of railway traffic and changing gaits in legged locomotion*, T2015/16, October 2015, TRAIL Thesis Series, the Netherlands

Brands, T., *Multi-Objective Optimisation of Multimodal Passenger Transportation Networks*, T2015/15, October 2015, TRAIL Thesis Series, the Netherlands

Ardiç, Ö., *Road Pricing Policy Process: The Interplay Between Policy Actors, the Media and Public*, T2015/14, September 2015, TRAIL Thesis Series, the Netherlands

Xin, J., *Control and Coordination for Automated Container Terminals*, T2015/13, September 2015, TRAIL Thesis Series, the Netherlands

Anand, N., *An Agent Based Modelling Approach for Multi-Stakeholder Analysis of City Logistics Solutions*, T2015/12, September 2015, TRAIL Thesis Series, the Netherlands

Hurk, E. van der, *Passengers, Information, and Disruptions*, T2015/11, June 2015, TRAIL Thesis Series, the Netherlands

Davydenko, I., *Logistics Chains in Freight Transport Modelling*, T2015/10, May 2015, TRAIL Thesis Series, the Netherlands