Efficient Optimization Methods for Freeway Management and Control

Zhe Cong

Cover design: Ying Li/李莹

Efficient Optimization Methods for Freeway Management and Control

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 maandag 30 november 2015 om 10.00 uur
door **Zhe CONG**,
Master of Science in Control Engineering, Huazhong University of Science and Technology,
geboren te Wuhan, Hubei, China.

This dissertation has been approved by the promotors:

Prof. dr. ir. B. De Schutter
Prof. dr. R. Babuška, M.Sc

Composition of the doctoral committee:

Rector Magnificus	chairperson
Prof. dr. ir. B. De Schutter	Technische Universiteit Delft, promotor
Prof. dr. R. Babuška, M.Sc	Technische Universiteit Delft, promotor

Independent members:

Prof. dr. S. Sacone	Università degli Studi di Genova
Prof. dr. T. Stützle	Université Libre de Bruxelles
Prof. dr. ir. E. van Berkum	Universiteit Twente
Prof. dr. ir. B. van Arem	Technische Universiteit Delft
Prof. dr. ir. C. Vuik	Technische Universiteit Delft

Research described in this thesis was supported by the China Scholarship Council (CSC), Delft Center for Systems and Control, the European COST Action TU1102, and by the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 257462 HYCON2 Network of Excellence.

TRAIL Thesis Series T2015/17, 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: Zhe Cong
E-mail: congzhefeb@gmail.com

ISBN 978-90-5584-197-4

Keywords: freeway networks, optimization, dynamic traffic routing, co-design of topology and control measures, UAV path planning

Copyright © 2015 by Zhe Cong

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission of the author.

Printed in the Netherlands

Acknowledgements

Finally, I did it! Before I started my Ph.D, I knew it would not be an easy journey; but now when I look back my entire Ph.D life, the difficulty is still beyond my prediction. However, I enjoy every moment in the past six years, no matter it is sweet or bitter, because I am not alone. There are so many people to guide me, to help me, and to accompany me to reach this destination. I owe them a lot of thanks!

First and foremost, I would like to thank my supervisors Prof. Bart De Schutter and Prof. Robert Babuška for their support and guidance. Bart can always give a fast and accurate feedback on my work. No matter how late I sent him an email related to my work, he was always willing to sacrifice his private time to answer it. He not only helped me to tackle all the problems I have on my research, but also provided personal support on many other things. Even sometimes I disappointed him, he still encouraged me. I really want to thank him for his patience. I am also very grateful to Robert, who is a kind and strict tutor. He gave me a lot of good ideas when I got stuck in my research. He also gave me constructive criticism that helps me to improve my programing skills and English writing. I sincerely appreciate his guidance from every aspect.

Further, I would like to thank my Ph.D committee members, Prof. Simona Sacone, Prof. Thomas Stützle, Prof. Eric van Berkum, Prof. Bart van Arem, and Prof. Kees Vuik, for their valuable time and suggestions provided to improve my thesis.

Then, I would like to thank Deheng Liu and Xiao Zhang, who have been my friends for almost twenty years. Without them, I would not be able to come to the Netherlands, and none of these amazing things would happen in my life. They unselfishly helped me with all kinds of preparations before my travel, and after I came here, I never feel longly because they are my family in this country. During my Ph.D period, their baby son, Ziyan, was born, and I was honored to be his godfather. I also would like to thank this little boy for the happiness that he brought to me.

Of course, I would not forget to thank my girlfriend Yang Yu, who enlightens my life in the dull windy and rainy days in the Netherlands. For me, maintaining a relationship is probably as difficult as doing Ph.D, but luckily, she is always willing to forgive the mistakes I made. She is like a mirror to me, from whom I can see my shortcomings, and become better.

I feel lucky that I made a lot of good friends here, Guang Ge, Xiaodong Guo, Daoxin Li, Chi Liu, Lu Nie, Ruimin Yang, Tianyi Zhang. All the trips we had and all the adventures we made are my precious memories eternally .

For my old friends in China and all over the world, Wenjun Ding, Ming Huang, Changlong Li, Mingsu Liu, Yun Liu, Jie Long, Shan Luo, Wenchao Mao, Xuebo Song, Xue Wu, Jingying Xu, Ying Xu, Chen Yang, Li Yu, Wanxing Zhang, Xiangyu Zhang, and Xianming Zhou, although we cannot get together as frequently as before, I believe that the bonds between us are still tight.

When working in TU Delft, I have good times with my colleagues. I would like to thank Noortje, Alfredo, Zul, Shu Lin, Yu Hu, and Yihui for good memories of different conferences. I would like to thank Mohammad, Yiming, Chengpu, Reinier, Amir, Anqi, Hai, Le, Shuai Yuan, Renshi, Esmaeli, and Dieky for those intense table football games. I would like to thank our secretaries, Kitty, Marieke, Heleen, and Kiran for being helpful and supportive. I would like to thank Farid, Shuai Liu, Soloman, Pieter, Yashar, Max, Jia, Jianfei, Zhou Zhao, and Juan Guo for your friendship.

In additional, I would like to thank: Andreas, for providing his Matlab codes and thesis materials; Bart and Kim, for helping me with translating the summary of my thesis into Dutch; Fankai, for helping me with developing the C codes; Chunman, for being my most loyal lunch partner; Xin, for supporting me like a sister; Liang and Peipei, for teaching me tennis; Lincy, for sharing your job experience; Duan, for the fun of playing Xbox games together; Yang Xu, Jie Xu, and Yajing Huang, for being my roommates, and the dinners that we had together.

Most importantly, I would like to use this opportunity to thank my parents for their unconditional support and belief. Although I had plenty of hard times during my Ph.D life, I never felt hopeless. Their faith that I would eventually do it was even stronger than mine. Although they were thousands of miles away, they gave me the strength that I needed. Their constant love is the reason that I can complete this thesis.

Zhe Cong

Delft, November 2015

Contents

Chapter 1

Introduction

1.1 Background

Since the last century, the number of vehicles using road systems has been steadily increasing all over the world due to population growth and economic prosperity. Unfortunately, this increase in demand has resulted in more unstable traffic conditions, more frequent traffic congestion, and longer traffic delays. Today, such problems have grown to an unbearable level, particularly in metropolitan areas. The following scene can be observed every working day: drivers push themselves into already crowded freeways to join others who have been trapped in traffic jams, and by doing so, those drivers also become part of the problem, causing huge loss for both the economy and environment. In order to prevent, or at least to alleviate traffic congestion, traffic management and control is urgently required.

Traffic management and control involves the implementation of strategies, policies, and technologies to improve the performance of traffic networks [55]. It consists of components such as infrastructure improvement, roadway operations and control, communications, detection and surveillance, emergency evacuations, and so on. The goal is to provide drivers with safe, reliable, and sustainable travel in a changing environment with varying demand, and, at the same time, to take social and environmental factors into account.

This thesis develops several management and control strategies to improve the performance of traffic networks, with a particular focus on freeway networks. More specifically, this thesis involves

1. dynamic route guidance for vehicles traveling in a freeway network;
2. co-design of the topology of a freeway network (e.g., by construction), and traffic control measures;
3. path planning for unmanned aerial vehicles to monitor traffic conditions.

Since the corresponding problems are too complex to be solved analytically, they are all formulated as optimization problems, and solved using optimization techniques. Usually such problem formulations in a large-scale freeway network will result in nonlinear and nonconvex optimization problems. The main contribution of this thesis is to find a way to solve these problems efficiently with a well-balanced trade-off between performance and computation speed. The considered approaches include reduction, approximation, and reformulation of the problems, development of new optimization algorithms (Ant Colony Optimization), and the hierarchical solution methods.

The rest of this chapter is organized as follows. Section 1.2 provides some basic concepts about freeway networks. Then, the objectives of the research presented in this thesis are given in Section 1.3. Finally, a general overview, including the outline and the main contributions of this thesis, is presented in Section 1.4.

1.2 Basic Concepts

This section gives some basic definitions and concepts of traffic flow theory and general freeway networks, which will be used in this thesis. More information about traffic flow theory and freeway networks can be found in [38] and [1], respectively.

1.2.1 Traffic Flow Theory

Traffic Flow Features

Three important variables are often used to describe the behavior of traffic flow over different locations and observation periods:

- **Flow** is the number of vehicles passing a given point of a roadway during a given time interval. It is equivalent to the term 'volume' in certain traffic engineering circles.
- **Density** is the number of vehicles occupying a given length of a roadway at a particular time instant.
- **Speed** can be averaged across space and time. If the average is taken at a specific location over a time interval, it is called time-mean speed, while if the average is taken at a specific time instant over a space interval, it is called space-mean speed.

An approximate relationship among these variables is: flow is equal to density times space-mean speed.

Fundamental Diagram

The Fundamental diagram is introduced to analyze the relationship between the basic traffic variables (flow, density, and speed), and it can yield two of the three variables at a specific point in space if the third one is given. Moreover, because three-dimensional curves are not easy to plot on a sheet of paper, various two-dimensional representations of the relationship are often used. Figure 1.1 gives an example of a diagram of flow versus density, and speed is given by the slope of the line connecting a particular point with the origin. It should be emphasized that these relationships are approximations, and only close to reality when measuring many vehicles.

Macroscopic Traffic Model

One of the goals of investigating traffic dynamics is to predict the future traffic states from some set of initial conditions, and some time-varying data. To make such a prediction, it would be necessary to know how each vehicle reacts to different circumstances in its environment. However, precisely predicting the detailed behavior of each individual vehicle is impossible. Therefore, an alternative is to make such predictions based on coarse data such

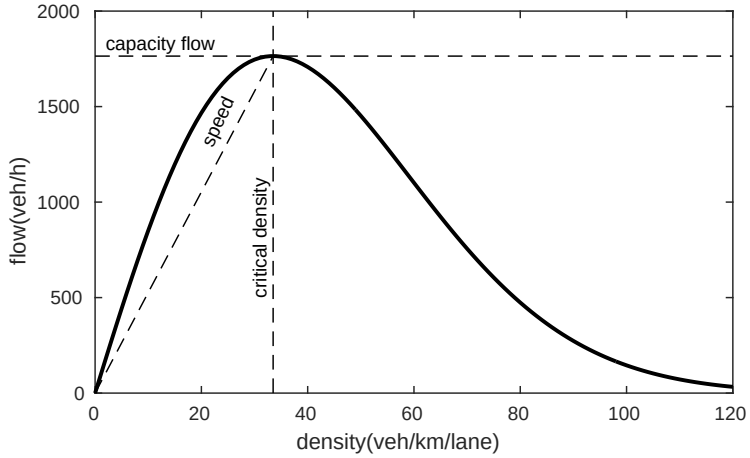


Figure 1.1: An example of the fundamental diagram of traffic behavior for a road with one lane. The slope of the line connecting the origin and a point on the fundamental diagram represents the average travel speed corresponding to that point. At the critical density, the capacity of the link is reached; if the density increases above the critical density, the resulting flow is below capacity.

as cumulative number of vehicles passing a road location. The best-known macroscopic traffic model is called the Lighthill-Whitham-Richards (LWR) model [92, 119], which is developed based on the fundamental diagrams introduced above.

1.2.2 Solutions for Reducing Congestion

Congestion mainly results from the fact that traffic demand approaches or exceeds the available capacity of freeways. Efficient freeway management and control aims at finding a balance between capacity and demand. The demand can vary significantly depending on the season of the year, the day of the week, and even the time of the day. On the other hand, the capacity is impacted by physical attributes of the freeway such as number of lanes, lane width, and degree of curvature, and by some other factors, such as weather, work zones, traffic incidents, or other events. In order to reduce congestion in freeway networks, two major solution methods are usually used : construction, or traffic control measures.

Construction

Construction refers to creating new freeways connections, or adding new lanes on existing roads. Construction often seems to be the first choice to deal with congestion, because it can provide a visible increase in capacity of freeway networks. However, construction may have several drawbacks. First of all, it could be rather expensive and time consuming. Second, it may temporarily cause even greater inefficiency because the construction sites may block traffic. Third, even if new construction is done, it may only reduce existing congestion rather than eliminate it completely because it is practically impossible to build sufficient capacity to satisfy future demand.

Traffic Control Measures

Since construction is often not feasible or insufficient to significantly reduce congestion, introducing traffic control measures is an alternative to address congestion. A short description of the traffic control measures that are most frequently used in freeway networks is given below.

- **Ramp metering** determines the flow rate at which vehicles enter the freeway, and is implemented via traffic signals placed at the on-ramp. The vehicles should stop when the light turns red, and they can pass when the light turns green. The purpose of ramp metering is to control the number of the vehicles that enter the network, and to influence the traffic densities on the mainstream roads in order to prevent a traffic jam or breakdown. Fixed-time ramp metering was adopted at first, but currently dynamic ramp metering is used more and more [107]. ALINEA [108] is one of the best known examples of a dynamic ramp metering strategy.

- **Variable speed limits** can be used to restrict the traffic speed on the freeway. The implementation involves speed limit signs placed over or besides the roads to display the maximum allowed speed for the given freeway stretches. The main purpose is to increase safety by lowering the speed limits upstream of congested areas [111, 124]. However, variable speed limits can also be used to improve the traffic flows [141].

- **Route guidance** uses dynamic route information panels or on-board devices to assist drivers in choosing the routes to their destinations. The original purpose is to inform the drivers about the current state of the traffic, e.g. travel time or queue lengths on different routes, to allow the drivers to take corresponding route choice decisions [18, 97]. This method can also be used to persuade drivers to change their route choice in order to obtain a traffic assignment that gives a more optimal traffic performance from the system point of view [75].

- Other traffic control measures include peak lanes that are only open during peak hours, bi-directional lanes that change their direction at different times of a day, based on the direction of the highest traffic demand [99].

1.3 Problem Statement

1.3.1 Objectives

Freeway management and control is used to influence traffic flows so as to prevent or alleviate traffic congestion and to provide drivers with efficient and safe travel, or more generally to improve the performance of the freeway network. From the network operation point of view, the overall performance evaluation usually depends on a variety of objectives, such as throughput, travel time, safety, fuel consumption, emissions, and so on. Therefore, it is almost impossible to analytically find the best management and control strategy in a large-scale freeway network. One promising way to find the optimal strategy is to formulate the traffic problem as an optimization problem with respect to multiple objectives, and then to solve the optimization problem using numerical optimization techniques.

A major challenge for this approach is that it is usually characterized by an extremely high computation burden, especially when online optimization is required. Since a freeway network is a comparatively fast system, it usually requires that the control signals can be determined in a fast way; however, the computation speed of solving an optimization problem depends on the size of the freeway network and on the complexity of the problem. Using faster computer processors is possible but rather expensive, and in general, it still will not tackle the issued. Therefore, this thesis aims at improving the network performance, as well as increasing the computational efficiency of the solution methods of the optimization problem for traffic management and control in freeway networks.

The approaches adopted include:

- Reducing the complexity the optimization problems, e.g., problem approximation or simplification;
- Developing new algorithms to solve the optimization problems, e.g., using artificial intelligence techniques;
- Dividing the optimization problem into several sub-problems, and solving them in a hierarchical way.

1.3.2 Methodology

An overview of different methodologies for management and control in a traffic network can be found in [38, 78, 110]. This section discusses the ones that are used in this thesis.

- **Optimal control** aims at obtaining a sequence of optimal control signals based on the system optimum conditions. For a given control period with the time horizon H_h , a sequence of control signals $u(0), u(1), \ldots, u(H_\mathrm{h}-1)$ is determined by minimizing a control objective function subject to constraints. Kotsialos et al. [87] used a nonlinear optimal control approach to generate the splitting rates of traffic flows and the ramp metering rates in a freeway network. Based on this method, Kotsialos and Papageorgiou [84] developed a software tool called Advanced Motorway Optimal Control.
- **Model Predictive Control (MPC)** can be considered to be optimal control applied in a rolling horizon scheme. The difference is that optimal control has an open-loop structure, while MPC adopts a closed-loop control approach. More specifically, at each control step k_c, MPC determines a sequence of control signals $u(k_\mathrm{c}|k_\mathrm{c}), u(k_\mathrm{c}+1|k_\mathrm{c}), \ldots, u(k_\mathrm{c}+H_\mathrm{p}-1|k_\mathrm{c})$ for the prediction period $[k_\mathrm{c}T_\mathrm{c}, (k_\mathrm{c}+H_\mathrm{p})T_\mathrm{c}]$, with $u(k_\mathrm{c}+j|k_\mathrm{c})$ the control signal at the control step $k_\mathrm{c}+j$ based on the information of the control step k_c, T_c the control time length, and H_p the prediction horizon. However, only the first sample of the control signal $u(k_\mathrm{c}|k_\mathrm{c})$ is applied to the system, and then the horizon is shifted to the next prediction period. MPC has been successfully applied to online traffic control measures in freeway networks (see [7, 71]).
- **Parameterized feedback control** aims at finding the optimal parameters of the controller, and the control signals are determined based on the current state of the system via a feedback control law at each control step. ALINEA [108] determines the ramp metering rates via a feedback control law with fixed control parameters. [147] introduces a dynamic framework for parameterized feedback control, containing two layers: the

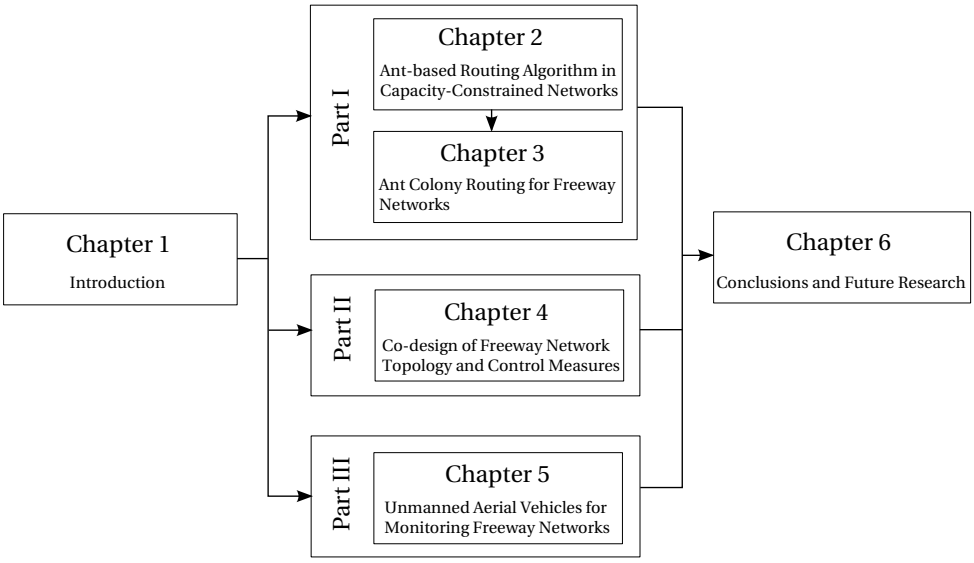


Figure 1.2: Structure of the thesis.

optimization layer and the control layer. At every control step, the optimization layer optimizes and updates the parameters of the control laws, and the control layer then determines the control signals via the control laws.

1.4 General Overview

1.4.1 Thesis Outline

This thesis contains six chapters (including this chapter), and it is divided into three parts. Each part deals with a different subtopic, and is written independently, which means that readers can read each part of the thesis separately. An overview of the relations between different parts and chapters is shown in Figure 1.2. The detailed content of each part is summarized as follows:

- Part I: Ant-based algorithms for dynamic traffic routing in freeway networks;
- Part II: Co-design of network topology and traffic control measures in freeway networks;
- Part III: Path planning for unmanned aerial vehicles to monitor the traffic conditions.

Part I contains two chapters. Chapter 2 first introduces a static routing algorithm for networks with capacity constraints, based on Ant Colony Optimization (ACO). The basic principle behind the use of ACO to solve a combinatorial optimization problem can be interpreted as artificial ants searching for the best path in an ant graph. Motivated by this feature of ACO, we propose the Ant Colony Optimization with Stench Pheromone (ACO-SP) algorithm to solve a network routing problem. ACO-SP uses a novel concept called stench pheromone to disperse ants to different paths, instead of letting ants converge on the same path in the

ant graph. The amount of stench pheromone deposited on each arc is calculated based on the number of ants on that arc. After ACO-SP terminates, the resulting assignment of ants in the ant graph is used to determine the splitting rates for flows in the network. Based on Chapter 2, Chapter 3 proposes the Ant Colony Routing (ACR) algorithm for solving the dynamic routing problem for freeway networks. In order to increase the computation speed, before applying the ACR algorithm, a network pruning step is first implemented to remove some "unnecessary" links and nodes from the original freeway network. After that, the ACR algorithm is applied in a Model Predictive Control (MPC) framework.

Part II introduces a co-design approach that jointly optimizes the network topology and the traffic control measures. Usually, this co-design approach has a problem because of different time scales: For a given design period, the network topology is fixed once determined, while the traffic control measures have to be adapted according to the time-varying traffic situations. Therefore, instead of optimizing the traffic control signals, parameterized control laws are used, and their parameters are optimized according to a pre-defined objective function. The co-design method involves a nonlinear, non-convex optimization problem with mixed-integer variables, which will be computationally expensive when dealing with a large-scale network and multiple control measures. In order to tackle this issue, we consider four different solution frameworks according to computational complexity, namely separate optimization, iterative optimization, bi-level optimization, and joint optimization.

Part III addresses a path planning problem involving unmanned aerial vehicles (UAVs) for monitoring freeway networks. Two distinct monitoring settings are considered: in the first setting, the UAVs have two flying modes — monitoring and traversing, and in the second setting, the UAVs only have one flying mode, which means that they can only monitor when hovering in the air. For the first setting, the monitoring problem is formulated as a periodical multiple rural postman problem, and solved using Mixed-Integer Linear Programming (MILP). For the second setting, the problem is formulated as a Markov Decision Process (MDP). However, since for large-scale traffic networks the standard MDP solution methods are limited by the memory size, three alternative solution methods, namely the fitted Q-iteration, the Model Predictive Control, and the parameterized control, are proposed.

Finally, the thesis is concluded in Chapter 6, in which each part of the thesis is summarized, and several recommendations for the future work are presented.

1.4.2 Contributions

The main contributions of this thesis are:

- We develop an ant-based algorithm to solve the dynamic traffic routing problem in freeway networks, which introduces two concepts, called stench pheromone and colored ants, and proposes a fully-dynamic way of using artificial ants to calculate the link cost while traffic is traveling on that link.

- We define a unified problem formulation for co-design of network topology and traffic control measures in a model-based optimization framework, where the network topology design and traffic control measures are jointly applied to a traffic model, and a monetary cost is used to evaluate the performance of the traffic network.

- We consider the mobile sensor monitoring problem in two different settings: the first

setting is formulated as a periodical multiple rural postman problem; the second setting is formulated as a Markov Decision Process (MDP).

Part I

Chapter 2

Ant-Based Routing Algorithm in Capacity-Constrained Networks

Ant Colony Optimization (ACO) is a powerful optimization heuristic for combinatorial optimization. This chapter introduces an ant-based algorithm called Ant Colony Optimization with Stench Pheromone (ACO-SP) to solve a routing problem in a network with capacity constraints, i.e., in a network where for each link the amount of flow cannot exceed a given upper bound. The basic idea is to map the network into an ant graph, which shares the same topology as the network, and then to use artificial ants to search for the best routes in the ant graph. The stench pheromone is a new concept introduced to repel ants when too many of them converge to the same arc. Through the regular pheromone and the stench pheromone, ants can be dispersed over the ant graph in an optimal manner. The routing problem is then solved by distributing the flow over the network according to the ant assignment determined by ACO-SP. Moreover, for a static routing problem, ACO-SP can be recast as a linear programming (LP) problem such that the routing problem can be solved by an LP method, which in general results in a lower computation time while achieving a similar performance as ACO-SP.

2.1 Introduction

Ants in the natural world often display intelligent collective behavior when seeking the shortest path between their nest and a source of food. This self-organizing behavior results from the fact that ants deposit pheromone trails on their way from the food source back to the nest. Pheromone is used as a medium of communication among ants to indirectly exchange information. Although this information exchange has a local scope, the collective behavior of the whole ant colony leads to satisfying a global goal. Deneubourg was the first to perform the double bridge experiments [40] in order to model the behavior of foraging ants. Based on these experiments, Dorigo initially proposed the idea of an ant-based metaheuristic approach in his PhD thesis [43] to solve combinatorial optimization problems in computer science. The class of algorithms that Dorigo introduced is called Ant Colony Optimization (ACO) [46], and has proven to be very successful in solving a wide variety of combinatorial optimization problems, such as traveling salesman problems [44, 45], scheduling problems [126], vehicle routing problems [59], and so on. An overview of papers covering representative ACO applications is given in [16].

This chapter considers using ACO to solve a routing problem in a network with capacity

constraints. The routing solution aims at determining an assignment of flows on the routes through the network with a least cost, e.g., total travel distance or total travel time. If no capacity is imposed on the links of the network, the problem is simplified as finding the shortest or the fastest route in the network, which can then be easily solved by the standard ACO algorithms.

AntNet [30] is one of the well-known ant-based routing algorithms, proposed for telecommunication networks. In AntNet, each node maintains a routing table, and an information table of the statistics about the traffic distribution over the network. There are two types of ants in AntNet, namely forward ants and backward ants. Forward ants are used as regular data packages that move from node to node in the network, and backward ants retrace the paths of forward ants in the opposite direction to update the routing table and the information table. The data package is then routed to the best path determined by the ants. However, AntNet is not suitable for solving the routing problem in some other types of networks, e.g., freeway networks. AntNet is an adaptive routing algorithm, and it directly puts artificial ants in the network. However, a freeway network is a more complex and slower system than a telecommunication network, and therefore it is impractical to embed ants (e.g., an ant-function car) into the freeway networks. The problem needs to be first solved in a separate ant graph, and then the solutions are implemented via traffic controllers in the freeway networks. Therefore, this chapter proposes a different ant-based dynamic routing algorithm for networks with capacity constraints, based on the work of biologists studying a special species of ants, called Lasius niger.

Dussutour et al. [49] investigated how Lasius niger ants move on a diamond-shaped bridge with two branches of equal length with limited capacity. They experimentally showed that when both branches of the bridge are wide enough, the majority of ants use the same branch. However, after reducing the width of both branches, the ants no longer converged to one branch, but instead almost a half of them moved to the other branch. Fourcassié et al. [56] interpreted this phenomena as that Lasius niger ants have the ability to optimally build and maintain the foraging trails for the colony in order to maximize the rate of food delivery to their nest. More specifically, if a route that ants travel along between a food source and the nest becomes congested, then the number of collisions between ants will increase, and, as a result, some ants are pushed to an alternative route by their opponents. This eventually results in a congestion-free distribution of ants in the experiment [49].

In order to actively disperse ants in the network as discussed above, we propose a concept called the *stench pheromone*. The stench pheromone has an opposite function to the regular pheromone used in the standard ACO algorithms: it can push ants away when there are too many ants on the same route. When no more ants can travel on the best (e.g., the shortest) route in the ant graph, they will start to search the second best, the third best, and so on. In this way, an optimal distribution of ants is achieved. Because of this concept, this ACO version is called as Ant Colony Optimization with Stench Pheromone (ACO-SP for short). In fact, the idea of using a pheromone with such a negative effect is not completely new. Montgomery and Randall [102] introduced a so-called anti-pheromone, which has the opposite effect to the regular pheromone. However, the goal of anti-pheromone differs from ours. It is used to prevent the ACO algorithm from converging too soon so as to avoid local optima when solving an optimization problem. A more similar concept to the stench pheromone is called pheromone repulsion [105], which is related to multiple-colony ant systems [59, 80]. In this kind of algorithm, ants are repulsed by the pheromones of the ants from other colonies. Because of the competition between different colonies, ants can find disjoint paths

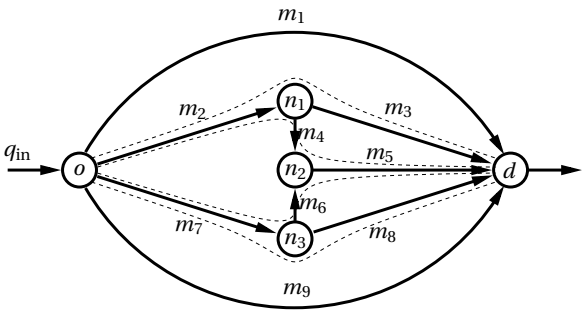


Figure 2.1: Illustration of how inflow q_{in} is distributed along multiple routes in a network. The dashed lines indicate the routing solutions.

in an ant graph. Compared to pheromone repulsion, the ACO-SP algorithm described in the thesis is simpler, yet can still effectively diversify the search. It has to be emphasized that in Chapter 3, we introduced a concept called colored ants, which can be considered as a kind of multiple-colony ant system. However, in our approach, ants feel neutral to the pheromones deposited by the ants from other colonies. The colored ants are only used for mapping a network with multiple destinations into an ant graph, where each color corresponds to one destination.

The rest of this chapter is structured as follows. The problem statement is introduced in Section 2.2. Then, the standard ACO algorithm is briefly recapitulated in Section 2.3, followed by the introduction of the new ACO-SP algorithm in Section 2.4. Section 2.5 discusses how to recast ACO-SP as a linear programming problem under certain conditions. In Section 2.6, a simulation-based case study is implemented to illustrate the use of ACO-SP for solving a routing problem in a simple road network. Finally, Section 2.7 concludes this chapter.

2.2 Problem Statement

This chapter only considers static routing problems; dynamic routing problems will be considered in the next chapter. It should be emphasized that ACO-SP may be overqualified for solving static routing problems. However, for illustration purposes, it is easier to consider a static routing problem to clearly explain the ACO-SP algorithm.

A general network can be modeled as shown in Figure 2.1, with a set $\mathscr{M}$ of links and a set $\mathscr{N}$ of nodes. For illustration purposes, this chapter only considers a network with one origin and one destination; the multi-origin multi-destination case will be considered in Chapter 3. The inflow of the network is denoted by q_{in}, and the flow on link $m \in \mathscr{M}$ is denoted by q_m. The total inflow of node $n \in \mathscr{N}$ is computed by:

$$Q_n = \sum_{m \in \mathscr{I}_n} q_m, \tag{2.1}$$

with $\mathscr{I}_n$ the set of incoming links of node n. Each outgoing link $m \in \mathscr{O}_n$ of node n is then characterized by:

$$q_m = \beta_{n,m} \cdot Q_n, \tag{2.2}$$

with $\beta_{n,m}$ the splitting rate that indicates the fraction of the total flow Q_n that is leaving via link m, and $\mathcal{O}_n$ the set of outgoing links of node n. The splitting rate is bounded by

$$0 \leqslant \beta_{n,m} \leqslant 1 \ , \tag{2.3}$$

and

$$\sum_m \beta_{n,m} = 1 \ . \tag{2.4}$$

Each link has a cost φ_m per unit flow. The static routing problem is solved by minimizing the objective function:

$$J = \sum_{m \in \mathcal{M}} \varphi_m \cdot q_m \ , \tag{2.5}$$

subject to the constraints (2.1)-(2.3).

The basic idea of using ACO-SP to solve a routing problem is to translate the network into an ant graph[1], mapping a node n into a vertex s or t, a link m into an arc (s,t), with the relationship $m = \ell(s,t)$ that indicates that a link m corresponds to an arc (s,t), and the link cost φ_m of each link m is translated into the length $L_{s,t}$ of each arc (s,t). The ACO-SP algorithm is used to minimize the objective function J in (2.5). Note that ACO-SP runs on the ant graph, not the network. The splitting rate $\beta_{n,m}$ is calculated based on the number of ants $y_{s,t}$ that have traveled each arc (s,t) with $m = \ell(s,t)$:

$$\beta_{n,\ell(s,t)} = \frac{y_{s,t}}{\sum_{t' \in \mathcal{N}_s} y_{s,t'}} \tag{2.6}$$

with $\mathcal{N}_s$ the set of vertices connected to vertex s. At the end, the flow on each link m is determined according to (2.2).

2.3 Standard Ant Colony Optimization

This section briefly reviews the standard ACO algorithms in Section 2.3.1, and explains the mechanism of one of the best-known ACO algorithms, the Ant System (AS), in Section 2.3.2.

2.3.1 Framework of ACO

The class of ACO algorithms has been developed to solve combinatorial optimization problems. In ACO, a combinatorial optimization problem is represented by a graph $\mathcal{G}(\mathcal{V},\mathcal{A})$, called the ant graph, which consists of a set $\mathcal{V}$ of vertices, and a set $\mathcal{A}$ of arcs connecting the vertices. A solution of the combinatorial optimization problem is an ordered set of solution components, with each solution component corresponding to a pair of vertices in the ant graph $\mathcal{G}$. Therefore, a solution can be considered to be a route over $\mathcal{G}$. To find a particular solution, an artificial ant a on $\mathcal{G}$ starts on an initial vertex, and moves from one vertex to another, adding the corresponding arc to its route r_a, until it reaches the terminal vertex.

[1] In order to avoid confusion, in this chapter the terms “links" and “nodes" refer to the network, and the terms “arcs" and “vertices" refer to the ant graph.

Then, the construction of the route r_a is completed, and r_a then represents a candidate solution found by ant a. The way that the terminal vertex is defined depends on the problem considered. For example, in the traveling salesman problem, the terminal vertex is the same as the initial vertex of ants, and in a control problem, the terminal vertex corresponds to the desired state of the system.

Two variables are associated with an arc (s,t): a pheromone variable $\tau_{s,t}$, and a heuristic variable $\eta_{s,t}$. The pheromone variable $\tau_{s,t}$ represents the knowledge acquired by ants about the optimal solutions over time, and the heuristic variable $\eta_{s,t}$ provides a priori information about the quality of the given solution component, i.e., the quality of moving from vertex s to vertex t. In general, a heuristic variable represents a short-term quality measure of each solution component, while a pheromone variable evaluates the quality of concatenating the respective solution components over a long term.

2.3.2 Ant System

Many ACO algorithms have been developed, e.g., Ant System (AS) [47], Max-Min Ant System [128], Rank-based Ant System [24], and Ant Colony System [44], most of them only differing in a few minor points to make them perform better on a specific type of problems. Among these algorithms, AS stands at the basis for most of the ACO algorithms later developed, including the ACO-SP algorithm introduced in the chapter.

The AS algorithm works as follows [47]. At the beginning, the pheromone variables $\tau_{s,t}$ are set to some initial value $\tau_0 > 0$, and the heuristic variables $\eta_{s,t}$ may be set to encode priori knowledge of the problem by favoring the choice of some vertices over others. Then, the AS algorithm keeps running in two loops: an inner loop, in which the solutions are constructed, and an outer loop, in which the pheromone levels are updated:

- **Inner loop**: Each ant a is put at some initial vertex, and each route r_a is initially empty. In each step, r_a is extended by adding arc (s,t) to it if ant a moves from vertex s to vertex t. The probability for ant a on vertex s to move to vertex t within its feasible neighborhood $\mathcal{N}_{s,a}$ is defined as:

$$p_a(t|s) = \frac{\tau_{s,t}^{\alpha}\eta_{s,t}^{\beta}}{\sum_{t' \in \mathcal{N}_{s,a}} \tau_{s,t'}^{\alpha}\eta_{s,t'}^{\beta}}, \quad \forall t \in \mathcal{N}_{s,a} \tag{2.7}$$

  with $\alpha > 0$ and $\beta > 0$ being user-defined parameters to control the influence of $\tau_{s,t}$ and $\eta_{s,t}$. The feasible neighborhood $\mathcal{N}_{s,a}$ of ant a on vertex s is the set of vertices that are connected to s and that have not yet been visited by ant a in the current inner loop. If the terminal vertex is reached, ant a stops searching, and route r_a is put into a set $\mathcal{S}_{\text{upd}}$ of candidate solutions that are constructed, and used for the pheromone update in the current iteration of the outer loop. Note that if $\mathcal{N}_{s,a} = \emptyset$, r_a cannot lead to a valid route. In that case, the construction of r_a is aborted, not used for pheromone update, and ant a stops as well.

- **Outer loop**: In each iteration, ant a deposits a pheromone trail $\Delta\tau_{s,t}(r_a)$ on arc (s,t) if arc (s,t) is a part of r_a. The pheromone trail $\Delta\tau_{s,t}(r_a)$ is computed based on a fitness

function $F(\cdot)$:

$$\Delta\tau_{s,t}(r_a) = \begin{cases} F(r_a)\,, & \text{if } (s,t) \in r_a \\ 0\,, & \text{otherwise} \end{cases} \tag{2.8}$$

The function $F(\cdot)$ assigns strictly positive values to each route r_a. In the ant graph, the better the route is, i.e., the shorter the route is, the higher the value of $F(\cdot)$ is. One way to implement $F(\cdot)$ is:

$$F(r_a) = \frac{K}{L_{r_a}}\,, \text{ with } L_{r_a} = \sum_{(s,t)\in r_a} L_{s,t} \tag{2.9}$$

with $L_{s,t}$ the length of arc (s,t), and $K > 0$ a constant. Then, the pheromone variable $\tau_{s,t}$ is updated as follows:

$$\tau_{s,t} \leftarrow (1-\rho_{\text{evap}})\tau_{s,t} + \sum_{r_a \in \mathscr{S}_{\text{upd}}} \Delta\tau_{s,t}(r_a)\,, \tag{2.10}$$

with $\rho_{\text{evap}} \in (0,1]$ the evaporation rate, which has the purpose of uniformly decreasing the pheromone values in order to avoid too rapid a convergence towards a sub-optimal solution.

When the difference in the pheromone levels between two consecutive iterations of the outer loop is smaller than a pre-defined threshold ϵ_τ, or when the iteration number of the outer loop reaches a specified maximum number $I_{\max}$, the AS algorithm terminates. The final solution is then extracted by selecting the vertices from the initial vertex to the terminal vertex. More specifically, at any vertex s, the next vertex t in the final route is selected by

$$t = \arg\max_{t'} \left(\tau_{s,t'}^{\alpha} \eta_{s,t'}^{\beta} \right)\,. \tag{2.11}$$

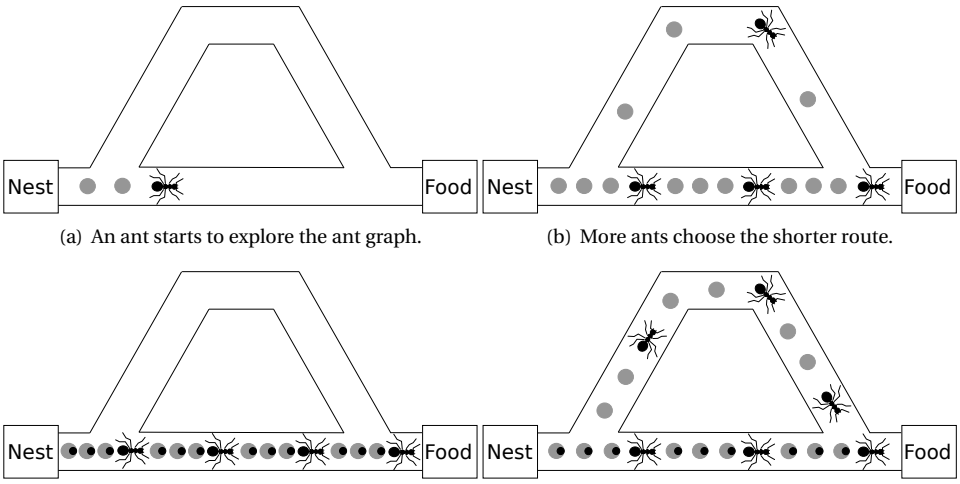
2.4 Ant Colony Optimization with Stench Pheromone

This section first presents the basic principles of the ACO-SP algorithm with an example in Section 2.4.1, then explains how to construct the stench pheromone function in Section 2.4.2, and finally describes ACO-SP in a mathematical way in Section 2.4.3.

2.4.1 Basic Principles

Let us illustrate the ACO-SP algorithm with an example, see Figure 2.2, which shows how ants move under the impact of both regular pheromone and stench pheromone. The ant graph has one initial vertex, the nest, one terminal vertex, the food source, and two arcs, with one being longer than the other. Suppose that the total number of ants is larger than the capacity on either arc, but smaller than the sum of capacities on both arcs. With this assumption, not all ants can travel on the same arc, but they have to split.

1. At the beginning, each ant starts to explore the ant graph, randomly selecting the shorter arc or the longer arc, and deposits the regular pheromone (indicated by the light grey dots) on the chosen arc. See Figure 2.2(a).



(a) An ant starts to explore the ant graph.

(b) More ants choose the shorter route.

(c) Too many ants converge to the shorter route.

(d) A part of the ants choose the longer route.

Figure 2.2: An example to illustrate how ants choose routes in the ACO-SP algorithm. The light grey dots indicate the regular pheromone, and the black dots indicate the stench pheromone.

2. As shown in Figure 2.2(b), when the number of ants is much lower than the capacity of the arc, only regular pheromone is deposited on that arc. So far, the process of ACO-SP is the same as that of the standard ACO algorithm [44, 47], and therefore more and more ants will gradually choose the shorter arc.

3. When the number of ants is approaching the capacity of the shorter arc, congestion will occur and as a result the performance (e.g., the speed of delivering food to the nest) will deteriorate. In this case, stench pheromone (indicated by the black dots) is deposited on that arc, resulting in a decrease of the total pheromone level (See Figure 2.2(c)). As a result, the probability of ants choosing the shorter arc accordingly decreases, and some ants will start to take the longer route.

4. Under the combined effect of the regular pheromone and the stench pheromone, the numbers of ants traveling on both arcs will eventually evolve towards some final values. Consequently, the network yields a steady-state optimal distribution of ants as shown in Figure 2.2(d).

Note that the stench pheromone is not deposited by ants, but is determined by some centralized entity according to the number of the ants on the arc. Therefore, the shape of the function that describes how much stench pheromone is deposited is designable. If the stench pheromone function has a strong impact, then the number of ants on each route will be small, while if the stench pheromone function has a weak impact, then the number of ants on each route will be large. Therefore, the stench pheromone function should be designed based on the requirements of the network managers. The construction of the stench pheromone function will be discussed in Section 2.4.2.

Generally speaking, regular pheromone can be considered as a reward for ants, while stench pheromone can be considered as a penalty for ants. From a control point of view,

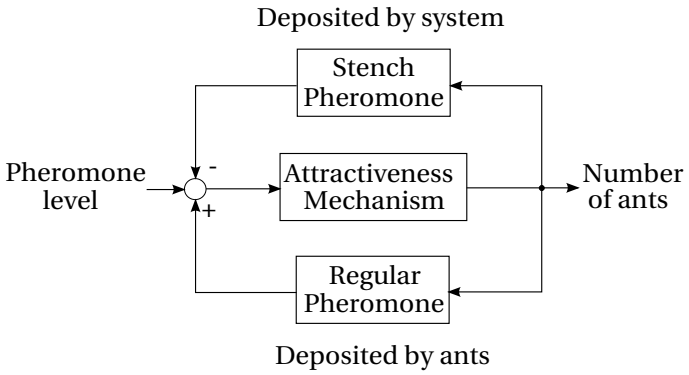


Figure 2.3: Illustration of the ACO-SP algorithm from a feedback point of view.

ACO-SP involves a feedback mechanism as shown in Figure 2.3. The total pheromone level on each arc results from the regular pheromone minus the stench pheromone, the number of ants on that arc is determined by the total pheromone level via the attractiveness mechanism, and the amount of the regular pheromone and the stench pheromone is calculated based on the number of ants.

As regards the convergence properties of the ACO-SP algorithm, we provide a proof for a simple case involving an ant graph with two arcs (see Appendix A). In this case, the updating pheromone process has four different modes:

- M1: No stench pheromone is deposited;
- M2: Stench pheromone is only deposited on arc 1;
- M3: Stench pheromone is only deposited on arc 2;
- M4: Stench pheromone is deposited on both arcs.

In Appendix A, we show that, for a specific range of parameters of the stench pheromone function, the process will transit among M1, M2, M3, and M4 in different ways, depending on the total numbers of ants and the thresholds for depositing the stench pheromone. However, no matter which mode transition sequence results, the pheromone levels on the two arcs will always converge.

2.4.2 Stench Pheromone Function

The deposit of the stench pheromone is activated by the capacity constraint, or by some predefined thresholds in terms of the numbers of ants on the arcs, depending on the problem considered. Moreover, the amount of stench pheromone deposited on arc (s,t) should correspond to the number $y_{s,t}$ of ants that have selected arc (s,t) as a part of their (valid) final route. The more ants choose arc (s,t), the more stench pheromone is deposited. The amount of stench pheromone deposited on arc (s,t) is calculated through a function $G_{s,t}$: $y_{s,t} \rightarrow G_{s,t}(y_{s,t})$, which has the following properties:

1. $G_{s,t}(0) = 0$, i.e., if no ants have selected link (s,t), no stench pheromone is deposited;

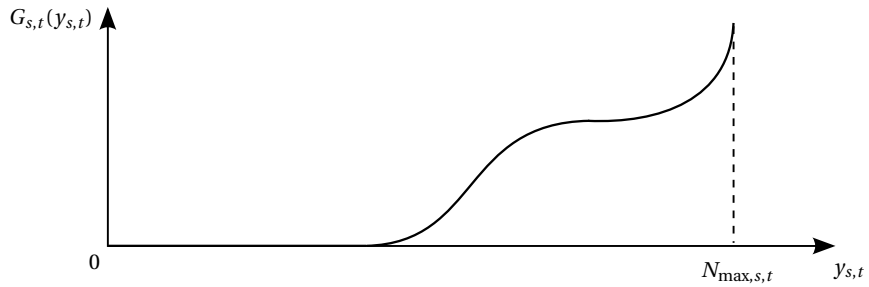


Figure 2.4: General shape of the stench pheromone function $G_{s,t}(\cdot)$.

2. $G_{s,t}$ is monotonically non-decreasing as $y_{s,t}$ increases;

3. $G_{s,t}$ can include one or more intermediate threshold levels, at which the value of $G_{s,t}$ increases significantly;

4. $G_{s,t}$ rises steeply as soon as $y_{s,t}$ reaches the capacity $N_{\max,s,t}$ of the arc.

Figure 2.4 shows the general shape of the stench pheromone function $G_{s,t}$. The specific form of $G_{s,t}$ depends on the network routing problem at hand.

One way to formulate $G_{s,t}$ is to use a piecewise affine (PWA) function :

$$G_{s,t}(y_{s,t}) = \begin{cases} P_{s,t,0} \cdot y_{s,t}, & \text{if } 0 \leqslant y_{s,t} < N_{s,t,1} \\ P_{s,t,1}(y_{s,t} - N_{s,t,1}) + B_{s,t,1}, & \text{if } N_{s,t,1} \leqslant y_{s,t} < N_{s,t,2} \\ \quad \vdots & \\ P_{s,t,j}(y_{s,t} - N_{s,t,j}) + B_{s,t,j}, & \text{if } y_{s,t} \geqslant N_{s,t,j} \end{cases} \tag{2.12}$$

with $N_{s,t,j}$ the jth threshold number of ants on arc (s,t), $P_{s,t,j} \geqslant 0$ the slope of the jth affine sub-function, and $B_{s,t,j}$ a constant to guarantee continuity of the stench pheromone function, which is

$$B_{s,t,i} = \begin{cases} P_{s,t,0} \cdot N_{s,t,1} \ , & \text{for } i = 1 \\ B_{s,t,i-1} + P_{s,t,i-1}(N_{s,t,i} - N_{s,t,i-1}) \ , & \text{for } i = 2, \cdots, j \end{cases} \tag{2.13}$$

The value of $N_{s,t,j}$ corresponds to the capacity of link $\ell(s,t)$, or some predefined threshold of flow on link $\ell(s,t)$.

Recall that the number $y_{s,t}$ of ants on each arc (s,t) in the ant graph is used to determine the splitting rate $\beta_{n,\ell(s,t)}$, which is the solution of the routing problem, on each link $\ell(s,t)$ in the network. Therefore, the regular pheromone and the stench pheromone should be deposited to eventually result in a well-balanced total pheromone level on each arc (s,t) that can lead to the minimal value of the objective function J in (2.5). For this purpose, the values of the parameters (e.g., the slopes $P_{s,t,j}$ and the threshold numbers $N_{s,t,j}$ in (2.12)) of the stench pheromone function need to be set properly.

In order to do that, we use a parameterization method. The basic idea of the parameterization method is to find the optimal parameters of the algorithms or the control laws (usually via solving an optimization problem) to optimize a pre-defined objective function. A more detailed description about this method will be introduced in Chapter 4, in which a

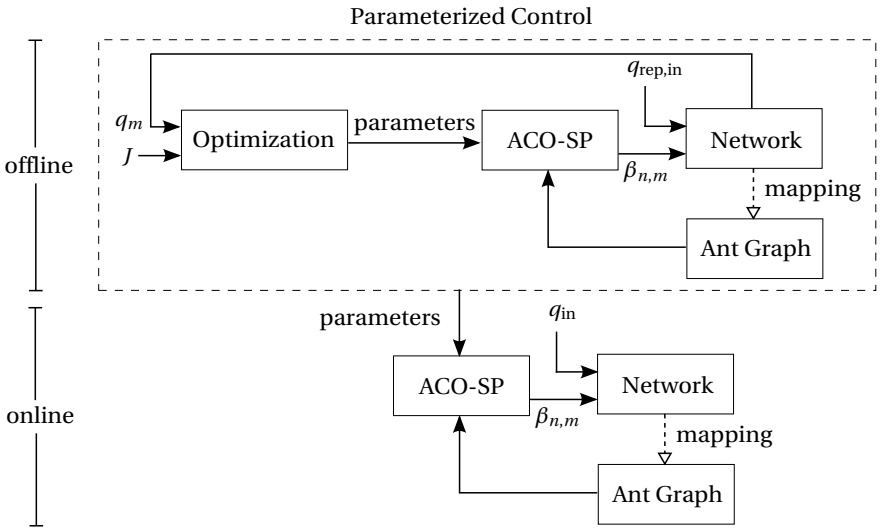


Figure 2.5: Schematic representation of using ACO-SP to solve the network routing problem by parameterized control. The dashed box indicates the parameterized control step. The ant graph itself is an input of ACO-SP.

parameterized traffic control approach will be used. Actually, the ACO-SP algorithm can be considered as a control law, and the splitting rate $\beta_{n,m}$ as the control signal.

Figure 2.5 illustrates how to use ACO-SP to solve the network routing problem by the parameterization method. There are two steps in this method: an offline parameterization step, and an online network routing step. In the offline step, we consider a representative inflow $q_{\text{rep,in}}$ entering the network. Then, an optimization approach is used to optimize the parameters of the stench pheromone function in ACO-SP in such a way that the objective function J is minimized. Next, the ACO-SP algorithm uses the parameters, together with the ant graph mapped from the network, to calculate the splitting rates. Subsequently, the splitting rates are applied to the network, and the newly generated flow on each link will be fed back to the optimization approach for a new iteration of optimizing J. The parameterized control step stops until the difference between the values of J in two consecutive iterations is smaller than a predefined tolerance, or the maximum number of iterations is reached. After that, we run the online step. The network is characterized by the real inflow q_{in}. We solve the routing problem by the ACO-SP algorithm, using the optimal parameters determined in the parameterization step.

Remark 2.1 For static routing problems, the representative inflow $q_{\text{rep,in}}$ can be selected the same as the network inflow q_{in}, while for dynamic routing problems, e.g., in a freeway network, the representative inflow $q_{\text{rep,in}}$ can be obtained via historical data, or based on a prediction model. □

2.4.3 Mathematical Formulation

In this section, the ACO-SP algorithm is formulated in a mathematical way. The framework of the ACO-SP algorithm (see Algorithm 2.1) is similar to the AS algorithm, including

Algorithm 2.1 The ACO-SP algorithm

Input: ant graph $\mathcal{G}(\mathcal{V},\mathcal{A})$, total number of ants $N_{\text{ants,total}}$, initial pheromone level τ_0, minimum pheromone level $\tau_{\min}$, evaporation rate ρ_{evap}, weight parameter α, maximum number of iterations $I_{\max}$, predefined tolerance ϵ_τ

$\tau_{s,t} \leftarrow \tau_0,\ \forall(s,t)$
repeat
  $\mathcal{S}_{\text{upd}} \leftarrow \emptyset$,
  $y_{s,t} \leftarrow 0,\ \forall(s,t)$
  for all ants $a \in \{1,2,\ldots,N_{\text{ants,total}}\}$ in parallel **do**
    $r_a \leftarrow \emptyset$
    initialize $\mathcal{N}_{s,a},\ \forall s \in V$
    $s \leftarrow$ initial vertex for ant a
    repeat
      select the next vertex t according to

$$p_a(t|s) = \frac{\left(\max\{\tau_{\min}, \tau_{s,t}\}\right)^{\alpha}}{\sum_{t' \in \mathcal{N}_{s,a}} \left(\max\{\tau_{\min}, \tau_{(s,t')}\}\right)^{\alpha}},$$

      $r_a \leftarrow r_a \cup \{(s,t)\}$,
      $\mathcal{N}_{s,a} \leftarrow \mathcal{N}_{s,a} \setminus \{t\}$,
      $s \leftarrow t$,
    until s is the terminal vertex **or** $\mathcal{N}_{s,a} = \emptyset$
    if s is the terminal vertex **then**
      $\mathcal{S}_{\text{upd}} \leftarrow \mathcal{S}_{\text{upd}} \cup \{r_a\}$
      $y_{s,t} \leftarrow y_{s,t} + 1,\ \forall(s,t) \in r_a$
    end if
  end for
  compute $G_{s,t}(y_{s,t}),\ \forall(s,t)$
  $\tau_{s,t,\text{prev}} \leftarrow \tau_{s,t}$
  update $\tau_{s,t}$:

$$\tau_{s,t} \leftarrow (1-\rho_{\text{evap}})\tau_{s,t} + \left(\sum_{r_a \in \mathcal{S}_{\text{upd}}} \Delta\tau_{s,t}(r_a)\right) - G_{s,t}(y_{s,t}),\ \forall(s,t)$$

until $i = I_{\max}$ **or** $|\tau_{s,t} - \tau_{s,t,\text{prev}}| \leqslant \epsilon_\tau,\ \forall(s,t)$
Output: $y_{s,t},\ \forall(s,t)$

the inner loop and the outer loop. In each iteration of the outer loop, the total pheromone level is updated by subtracting the amount of the stench pheromone from the regular pheromone. The pheromone update equation (2.10) is therefore modified as follows:

$$\tau_{s,t} \leftarrow (1-\rho_{\text{evap}})\tau_{s,t} + \left(\sum_{r_a \in \mathscr{S}_{\text{upd}}} \Delta\tau_{s,t}(r_a)\right) - G_{s,t}(y_{s,t}) \ . \tag{2.14}$$

By adopting this modification, the pheromone levels on the arcs may become negative. In that case, the probability $p_a(t|s)$ would not be well defined. Therefore, (2.7) should be modified as well:

$$p_a(t|s) = \frac{\left(\max\{\tau_{\min}, \tau_{s,t}\}\right)^{\alpha}}{\sum_{t' \in \mathscr{N}_{s,a}} \left(\max\{\tau_{\min}, \tau_{s,t'}\}\right)^{\alpha}}, \tag{2.15}$$

with $\tau_{\min} > 0$ a parameter that prevents the denominator of (2.15) from becoming zero. In (2.15), the heuristic variable $\eta_{s,t}$ is disregarded, assuming that no information about the quality of each arc is available a priori. This is implemented by setting all heuristic variables equal to one.

The termination conditions of the ACO-SP algorithm are the same as the ones of the AS algorithm. When ACO-SP terminates, the output of the algorithm is the *numbers of ants* $y_{s,t}$ on each arc (s,t).

2.5 Routing by Linear Programming

This section presents a linear programming (LP) method motivated by ACO-SP. The ACO-SP algorithm can be recast as an LP method if both of the following conditions are satisfied:

- **Condition (a)**: the link cost in the network is constant;
- **Condition (b)**: the stench pheromone function in ACO-SP is a convex piecewise affine (PWA) function.

Condition (a) always holds for static routing problems, and for dynamic routing problems, it also holds in a quasi-static method (see Section 3.3.3), in which the link cost is considered as fixed between every two control steps if the control time interval is not too large. Moreover, Condition (b) can usually be satisfied too as the stench pheromone function is constructed by the designer. In this way, the routing problem is formulated as an LP problem. Note that in this section, the ant graph is not used, so all the variables are defined for the network.

The primary objective is to minimize the total cost, which is defined as follows:

$$\min \ J_{\text{cost}} = \min_{q_m} \sum_{m \in \mathscr{M}} \varphi_m \cdot q_m \tag{2.16}$$

with q_m the flow on link m, φ_m the cost of link m, and $\mathscr{M}$ the set of links in the network. This minimization problem is constrained by:

$$\sum_{m \in \mathscr{O}_o} q_m = q_{\text{in}}, \tag{2.17}$$

$$\sum_{m\in\mathscr{I}_n} q_m = \sum_{m'\in\mathscr{O}_n} q_{m'},\ \forall n\in\mathscr{N}, \tag{2.18}$$

$$q_m \leqslant q_{\max,m},\ \forall m\in\mathscr{M}. \tag{2.19}$$

with q_{in} the incoming flow of the network, o the origin of the network, $q_{\max,m}$ the maximum flow on link m, $\mathscr{I}$ and $\mathscr{O}$ the sets of incoming and outgoing links respectively.

Recall that the stench pheromone function in ACO-SP is used to penalize that too many ants converge to the same arc. In the LP approach, it is considered as a penalty function $J_{\text{pen},m}$ with a PWA form:

$$J_{\text{pen},m} = \begin{cases} P_{m,0}q_m, & \text{if } 0 \leqslant q_m \leqslant Q_{m,1} \\ P_{m,1}(q_m - Q_{m,1}) + B_1, & \text{if } Q_{m,1} < q_m \leqslant Q_{m,2} \\ \quad\vdots & \\ P_{m,j}(q_m - Q_{m,j}) + B_j, & \text{if } Q_{m,j} < q_m \leqslant q_{\max,m} \end{cases} \tag{2.20}$$

with $Q_{m,i},\ i = 1,2,\ldots,j$, the predefined thresholds for the flow on link m, $P_{m,i}$ the slope of the ith affine sub-function, and B_i a constant to guarantee continuity of the function $J_{\text{pen},m}$, which is defined as:

$$B_i = \begin{cases} P_{m,0}Q_{m,1}, & \text{for } i = 1 \\ B_{i-1} + P_{m,i-1}(Q_{m,i} - Q_{m,i-1}), & \text{for } i = 2,\cdots,j \end{cases} \tag{2.21}$$

The penalty function $J_{\text{pen}} = \sum_{m\in\mathscr{M}} J_{\text{pen},m}$ is another objective to be minimized. If $J_{\text{pen},m}$ is a *convex* PWA function, then minimizing J_{pen} can be recast in an LP format by introducing an additional optimization variable g_m:

$$\min J_{\text{pen}} = \min_{q_m,g_m} \sum_{m\in\mathscr{M}} g_m \tag{2.22}$$

subject to

$$g_m \geqslant P_{m,0}q_m,\ \forall m\in\mathscr{M} \tag{2.23}$$

$$g_m \geqslant P_{m,1}(q_m - Q_{m,1}) + B_1,\ \forall m\in\mathscr{M} \tag{2.24}$$

$$\vdots$$

$$g_m \geqslant P_{m,j}(q_m - Q_{m,j}) + B_j,\ \forall m\in\mathscr{M} \tag{2.25}$$

It is easily to verify that the LP problem (2.22)–(2.25) amounts to minimizing J_{pen}.

By combining the two objectives, the LP routing problem is formulated as:

$$\min_{q_m,g_m}\ \alpha_1 J_{\text{cost}} + \alpha_2 J_{\text{pen}} \tag{2.26}$$

with $\alpha_1,\alpha_2 > 0$ the weight parameters, subject to the constraints (2.17)–(2.19) and (2.23)–(2.25). This LP problem can be solved using one of the many available algorithms for linear programming (see e.g., [112, Chapter 1] or [39]). In this way, the optimal flows q_m^* in the network are determined, yielding a balanced trade-off between minimizing the total cost and avoiding congestion.

Remark 2.2 However, if $J_{\text{pen},m}$ is a *non-convex* PWA function, we obtain a so-called mixed-integer linear programming (MILP) problem [11]. As will be shown next.

First, the penalty function $J_{\text{pen},m}$ is rewritten as a recursive function:

$$\begin{aligned} &g_{m,0} = P'_{m,0} q_m, \text{ if } 0 \leqslant q_m \leqslant Q_{m,1} \\ &g_{m,i} = g_{m,i-1} + P'_{m,i}(q_m - Q_{m,i}), \text{ if } Q_{m,i} \leqslant q_m \leqslant Q_{m,i+1}, \;\; i = 1, \ldots, j \end{aligned} \tag{2.27}$$

with $Q_{m,j+1} = q_{\max,m}$, and $P'_{m,i}$ a new parameter for slope, which can be obtained according to (2.20). Then, each condition $q_m \geqslant Q_{m,i}$ is associated with a binary logical variable $\delta_{m,i} \in \{0,1\}$ such that

$$[\delta_{m,i} = 1] \Leftrightarrow [q_m \geqslant Q_{m,i}] \,, \tag{2.28}$$

where "$\Leftrightarrow$" means "if and only if". It is easy to verify that (2.28) is equivalent to

$$q_m \leqslant Q_{m,i} - (Q_{m,i} - q_{\max,m})\delta_{m,i} \tag{2.29}$$

$$q_m \geqslant Q_{m,i} - Q_{m,i}(1 - \delta_{m,i}) \tag{2.30}$$

Then (2.20) can be rewritten as:

$$J_{\text{pen},m} = \delta_{m,0} P'_{m,0} q_m + \sum_{i=1}^{j} \delta_{m,i} \left(P'_{m,i}(q_m - Q_{m,i}) \right) \tag{2.31}$$

The term $\delta_{m,i} q_m$ can be replaced by an auxiliary real variable $z_{m,i} = \delta_{m,i} q_m$, which can be expressed as:

$$z_{m,i} \leqslant q_{\max,m} \delta_{m,i} \,, \tag{2.32}$$

$$z_{m,i} \geqslant 0 \,, \tag{2.33}$$

$$z_{m,i} \leqslant q_m \,, \tag{2.34}$$

$$z_{m,i} \geqslant q_m - q_{\max,m}(1 - \delta_{m,i}) \,. \tag{2.35}$$

So (2.31) is reduced to a linear equation:

$$J_{\text{pen},m} = P'_{m,0} z_{m,0} + \sum_{i=1}^{j} P'_{m,i}(z_{m,i} - Q_{m,i}\delta_{m,i}) \,. \tag{2.36}$$

subject to the linear constraints (2.29), (2.30), and (2.32)–(2.35). Note that the optimization variables in (2.36) include continuous variables q_m and $z_{m,i}$, and also binary variables $\delta_{m,i}$. In this way, the problem becomes an MILP problem, which can be solved efficiently by several existing state-of-the-art commercial and free solvers, such as CPLEX, Gurobi, Xpress-MP, or GLPK [3, 93]. □

2.6 Case Study

The case study considered here chooses a traffic scenario to illustrate the use of ACO-SP to solve a routing problem. Note however that ACO-SP can be used for solving other general routing problems. The case study network is chosen to be simple with only two essential

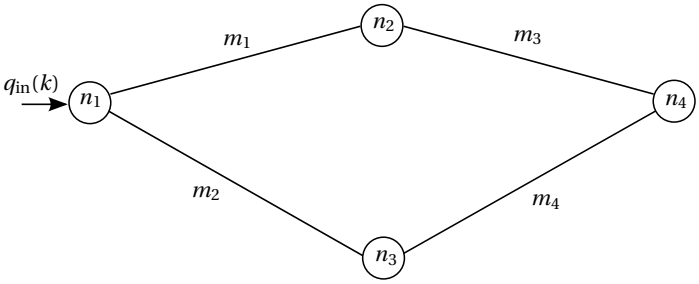


Figure 2.6: A simple network with one origin, one destination, and four links.

routes. For a traffic scenario, the network in general has a time-variant inflow. Therefore, we consider a discrete-time setting, where in each simulation period $[kT,(k+1)T]$ for $k = 0,1,\ldots$, with T the length of the simulation period, the traffic inflow and the traffic state on each link is fixed. In this way, we still solve a static routing problem in each simulation period. Moreover, in order to evaluate the performance and computational efficiency, ACO-SP is compared with another traffic routing method, namely the non-linear optimal control approach [87]. All simulations are programmed in Matlab by using a desktop computer with the Linux OS, and an Intel(R) Core(TM) 2 Duo CPU with 3.00 GHz and 4GB RAM.

2.6.1 Network set-up

The network of the case study is shown in Figure 2.6. The lengths of m_1 and m_3 are 2.5 km, and the lengths of m_2 and m_4 is 5 km. Each link is divided into several segments, with the segment length $L_1 = L_2 = L_3 = L_4 = 0.5$ km. The number of lanes is $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$, and the capacity is $C_1 = C_2 = C_3 = C_4 = 2000$ veh/h.

The traffic condition on each link is described by the fundamental diagram (see Figure 1.1), which can be mathematically formulated as:

$$v_{m,i}(k) = v_{\text{free},m}\exp\left(-\frac{1}{a_m}\left(\frac{\rho_{m,i}(k)}{\rho_{\text{crit},m}}\right)^{a_m}\right) \tag{2.37}$$

$$q_{m,i}(k) = \rho_{m,i}(k)\, v_{m,i}(k)\lambda_m \tag{2.38}$$

with $q_{m,i}(k)$, $v_{m,i}(k)$, and $\rho_{m,i}(k)$ the outflow, the speed, and the density of traffic on segment i of link m at time step k, $v_{\text{free},m}$ the free-flow speed, $\rho_{\text{crit},m}$ the critical density, and a_m the model parameter. The values are set as $v_{\text{free},m} = 50$ km/h, $\rho_{\text{crit},m} = 30$ veh/h/lane, and $a_m = 1.867$. Moreover, to follow the conservation law, the density of a segment at each time step equals the density of that segment at previous time step plus the inflow from the upstream segment, minus the outflow of the segment itself:

$$\rho_{m,i}(k+1) = \rho_{m,i}(k) + \frac{T}{L_m\lambda_m}(q_{m,i-1}(k) - q_{m,i}(k)) \tag{2.39}$$

with T the length of simulation time step, and it is set as $T = 60$ s. The inflow $q_{m,0}(k)$ of the virtual upstream segment is determined by the splitting rate:

$$q_{m,0}(k) = \beta_{o,m}(k)\, q_{\text{in}}(k) \tag{2.40}$$

with $q_{\text{in}}(k)$ the inflow of the network, which is set as $q_{\text{in}}(k) = 1800$ veh/h. We consider three different simulation periods, lasting 0.5 hour, 1 hour, and 1.5 hours respectively. The link cost is chosen as the travel time on each link:

$$\varphi_m(k) = \sum_{i=1}^{N_m} \frac{L_m}{v_{m,i}(k)} \tag{2.41}$$

with N_m the number of segments of link m. Therefore, the objective function at each simulation step k is then:

$$J(k) = \sum_{m \in \mathcal{M}} \varphi_m(k) \cdot q_m(k) \,. \tag{2.42}$$

2.6.2 ACO-SP settings

Note that although the case study network has four links, links m_1 and m_3, and links m_2 and m_4 can be mapped into one arc respectively when translating the network into the ant graph, because links m_3 and m_4 are the only outgoing link of links m_1 and m_2, and links m_3 and m_4 have the same number of lanes and the same capacity as links m_1 and m_2. So, we map links m_1 and m_3 into arc (s_1, t_2), and links m_2 and m_4 into arc (s_1, t_3).

The parameters of the ACO-SP algorithm are set as follows: the total number of ants is $N_{\text{ants,total}} = 5000$, the evaporation rate is $\rho_{\text{evap}} = 0.1$, the initial value of pheromone is $\tau_0 = 10$, the lower bound of pheromone is $\tau_{\min} = 3$, the model parameter $\alpha = 1$, the maximum number of the outer loop of ACO-SP is $I_{\max} = 1000$, the maximum number of simulation-optimization iterations is $N_{\text{iter}} = 10$, the tolerance for pheromone is $\epsilon_\tau = 10^{-6}$, and the tolerance for the splitting rate is $\epsilon_\beta = 10^{-3}$.

The stench pheromone function is defined as a PWA function. According to the fundamental diagram, traffic congestion may occur if the critical density is exceeded. Therefore, the threshold number $N_{s,t}$ of ants on arc (s,t) corresponds to the critical density of the link $m = \ell(s,t)$ in the network:

$$N_{s,t} = \gamma_{s,t} \rho_{\text{crit},\ell(s,t)} L_{\ell(s,t)} \lambda_{\ell(s,t)} \tag{2.43}$$

with $\gamma_{s,t} > 0$ the weight parameter for the threshold number $N_{s,t}$. The smaller $\gamma_{s,t}$ is, the easier the stench pheromone is activated. Then, the stench pheromone function is formulated as:

$$G_{s,t}(y_{s,t}) = \max(0, P_{s,t}(y_{s,t} - N_{s,t})) \,. \tag{2.44}$$

The slopes $P_{1,2}$ and $P_{1,3}$, and the weights $\gamma_{1,2}$ and $\gamma_{1,3}$ are optimized using the parameterized control method by considering two representative network inflows: $q_{\text{rep,in},1} = 1500$ veh/h, and $q_{\text{rep,in},2} = 2000$ veh/h. In this case study, we choose the genetic algorithm as the optimization method in parameterized control, which is implemented via the `ga` function of the Matlab Global Optimization Toolbox.

2.6.3 Simulation results

The results of optimizing the parameters of the stench pheromone function are $P_{1,2} = 4.4492$, $P_{1,3} = 3.5438$, $\gamma_{1,2} = 18.0956$, and $\gamma_{1,3} = 25.1981$.

Table 2.1: Comparison of the results of ACO-SP and the non-linear optimal control approach ("NA" indicates not applicable)

Methods	Simulation period [h]	TTS [veh· h]	Computation time [s]	
			For optimizing parameters	For solving routing problem
ACO-SP	0.5	204.1965	1222.1	682.5
	1.0	390.6310		1388.6
	1.5	605.4330		1997.4
Optimal control	0.5	198.6345	NA	441.6
	1.0	404.4612		2670.2
	1.5	586.7164		16213.7

The results of the routing problem are shown in Table 2.1. Since ACO is a stochastic algorithm, in fact we have run the simulation five times to compare the difference between each result set. Because the network has such a tiny size, the variability of the results is small, so we only show one specific set of the results here. Note that the non-linear optimal control approach of [87] is a comprehensive traffic control strategy, considering the use of ramp metering, route guidance, and freeway-to-freeway control measures. However, in order to compare it with ACO-SP, this case study only extracts the part involving the solution of the routing problem, with the objective of minimizing the total time spent (TTS). According to [87], this problem is solved by considering it as a discrete-time optimal control problem:

$$\min J = \min_{\beta_{n,m}(k)} T \sum_{k=1}^{K} \sum_{m \in \mathscr{M}} \sum_{i=1}^{N_m} \rho_{m,i}(k) L_m \lambda_m \tag{2.45}$$

subject to the constraints (2.37)–(2.40).

According to Table 2.1, the performance in terms of TTS resulting for the two methods is similar in this case study. More specifically, ACO-SP works better than the non-linear optimal control approach in the case that the simulation period is 1 hour, while the non-linear optimal control approach performs better in the other two cases. Comparing the computation time, ACO-SP has an offline step for optimizing the parameters of the stench pheromone function. This offline step only needs be executed once for the same representative inflow $q_{\text{rep,in}}$ in the same network. For solving the routing problem, ACO-SP is faster than the non-linear optimal control, especially in the third case.

The non-linear optimal control optimizes the splitting rates in a direct way; thus the number of the optimization variables is[2] $(N_{\text{link}} - N_{\text{node}}) \cdot K$, with N_{link} the number of links in the network, N_{node} the number of nodes in the network, and K the total number of time steps. Therefore, the complexity of the problem increases as the simulation period becomes longer. On the other hand, ACO-SP is a parameterized method, which aims at finding optimal parameters of the stench pheromone function to deposit an appropriate amount of stench pheromone on each arc. Therefore, the number of optimization variables will be much less. For instance, in this case study, the stench pheromone function is a PWA function, and the optimization variables are the slopes $P_{s,t}$ and the weight parameters $\gamma_{s,t}$ of

[2] Due to the constraint $\sum_{m \in \mathscr{O}_n} \beta_{n,m}(k) = 1$, for each node n, the number of the splitting rates to be optimized is $|\mathscr{O}_n| - 1$, with $\mathscr{O}_n$ the set of outgoing links of node n, and $|\cdot|$ the cardinality of a set.

the threshold numbers of ants. In this case, the number of the optimization variables is $2N_{\text{ants,thresh}} \cdot N_{\text{link}}$, with $N_{\text{ants,thresh}}$ the threshold number of ants for the stench pheromone function. Since the stench pheromone function is designable, the number $N_{\text{ants,thresh}}$ is in general a much smaller number than K. Therefore, compared to non-linear optimal control, ACO-SP is a more efficient algorithm to yield a well-balanced trade-off between performance and computation speed.

2.7 Conclusion

In order to solve the routing problems in a network with capacity constraints, a variant of the Ant Colony Optimization (ACO) method is proposed, called Ant Colony Optimization with Stench Pheromone (ACO-SP). For illustration purpose, we only consider using ACO-SP for the static routing problems in this chapter, and the dynamic routing problems will be introduced in the next chapter.

ACO-SP uses two conflicting types of pheromone, i.e., the regular pheromone and the newly introduced stench pheromone, to determine an optimal assignment of ants in the ant graph, so as to determine the splitting rates for flows in the network, with the objective of minimizing the total travel cost. Moreover, if Conditions (a) and (b) in Section 2.5 are satisfied, ACO-SP can be recast as a linear programming (LP) approach. To illustrate ACO-SP, a simulation-based case study in a traffic scenario is implemented on a simple network. The simulation result shows that ACO-SP can provide a well-balanced trade-off between performance and computation speed.

Chapter 3

Ant Colony Routing for Freeway Networks

Dynamic traffic routing refers to the process of (re)directing vehicles at junctions in a traffic network according to the traffic conditions. Traffic management centers can determine desired routes for drivers in order to optimize the performance of the traffic network by dynamic traffic routing. However, a traffic network may have thousands of links and nodes, resulting in a large-scale and computationally complex nonlinear, non-convex optimization problem. To solve this problem, Ant Colony Routing (ACR) is introduced in this chapter. ACR solves the dynamic traffic routing problem by applying an extended version of the Ant Colony Optimization with Stench Pheromone (ACO-SP) algorithm in a Model Predictive Control (MPC) framework. Moreover, in order to reduce the complexity of the problem, a network pruning step is implemented before the ACR step, to remove some links and nodes from the traffic network. To illustrate the effectiveness of this algorithm, the proposed method is tested in a simulation-based case study involving the Walcheren area in the Netherlands.

Research work of this chapter has been published in [35].

3.1 Introduction

Dynamic traffic routing [78] is an effective traffic management and control method that guides drivers to their route according to current (and future) traffic conditions when several alternative routes exist to their destination. In dynamic traffic routing, the notions system optimum and user equilibrium play an important role. According to [140], the system optimum is achieved when the vehicles are guided such that the total travel costs of all drivers (typically the total travel time) are minimized, while the user equilibrium means that on all alternative routes used, the costs are equal and minimal, and lower than those on the routes that are not used. In general, the system optimum and the user equilibrium are two conflicting objectives. The system optimum does not necessarily minimize the travel cost of every individual driver, so some drivers may have higher cost than the others, and in the user equilibrium the collective objective will not necessarily be optimized.

A broad literature exists on the topic of the dynamic traffic routing [87]. Papageorgiou and Messmer [106] have proposed a theoretical framework for route guidance in traffic networks. Three different traffic control problems are formulated: an optimal control problem w.r.t. the system optimum, an optimal control problem w.r.t. the user equilibrium, and a feedback control problem w.r.t. the user equilibrium. Wang et al. [138] have developed a predictive feedback controller based on the predicted travel time, which is the time that

drivers will experience when they drive along the give route. The results show that the predictive feedback control approach can yield nearly optimal splitting rates, and it is robust for several cases, e.g., incorrectly predicted demand, or an unpredicted incident. Kotsialos et al. [87] formulated the dynamic traffic routing problem as a discrete-time optimal control problem w.r.t. the system optimum, and used a numerical non-linear optimization algorithm to solve it. Other work includes e.g. analyzing the drivers' reaction to the provided guidance information [113, 114], and generating the route guidance instructions by using real-time measurements and short-term predictions of traffic situations [21].

The objective of this chapter is twofold:

1. Finding an optimal routing solution to reduce the travel cost (total time spent (TTS) in this chapter[1]) in the freeway network;
2. Controlling the number of vehicles on each link in the traffic network, especially in some sensitive zones, e.g., near hospitals and schools, in order to improve safety and to reduce noise and pollution.

For this purpose, this chapter introduces an Ant Colony Routing (ACR) algorithm, based on the Ant Colony Optimization with Stench Pheromone (ACO-SP) algorithm proposed in Chapter 2. Generally speaking, using the regular ACO algorithm for dynamic traffic routing suffers from four main issues:

1. Ants in ACO only strive for the user equilibrium, while traffic management has global objectives;
2. Ant graphs have no limiting capacities on arcs, while traffic networks are constrained by link capacity;
3. Ants typically have a common destination, while each vehicle in a traffic network has its own pre-determined destination;
4. The length of each arc in ant graphs is fixed and static, while the link costs in traffic networks dynamically depend on the time-varying traffic conditions.

The first two issues can be easily tackled by the original ACO-SP algorithm, where the stench pheromone is used to disperse ants over the ant graph for the sake of a system-wide objective, and the capacity constraints, as well as other limitations on the numbers of vehicles on links, can be addressed by properly defining the stench pheromone function. In order to tackle the other two issues, ACO-SP has to be extended especially for traffic scenarios. First of all, a new concept called colored ants is added, in which each color is assigned to a corresponding destination of the traffic network, and colored ants are only sensitive to the pheromone of their own color. Moreover, future traffic information is predicted by traffic simulation models, and is taken into account for the ant graph as well as the current traffic information.

The ACR algorithm applies the extended version of ACO-SP[2] within a Model Predictive Control (MPC) framework [95, 118]. However, before that, a network pruning step is implemented. This is because for a large-scale traffic network, some of the links and nodes compose routes that could be relatively long such that they are unlikely to be chosen by drivers.

[1] Any other cost, e.g., tolls, fuel consumption, and emissions, can be added to the travel cost.

[2] For the sake of compactness, in the rest of this chapter, the term ACO-SP refers to the extended version of the original ACO-SP algorithm.

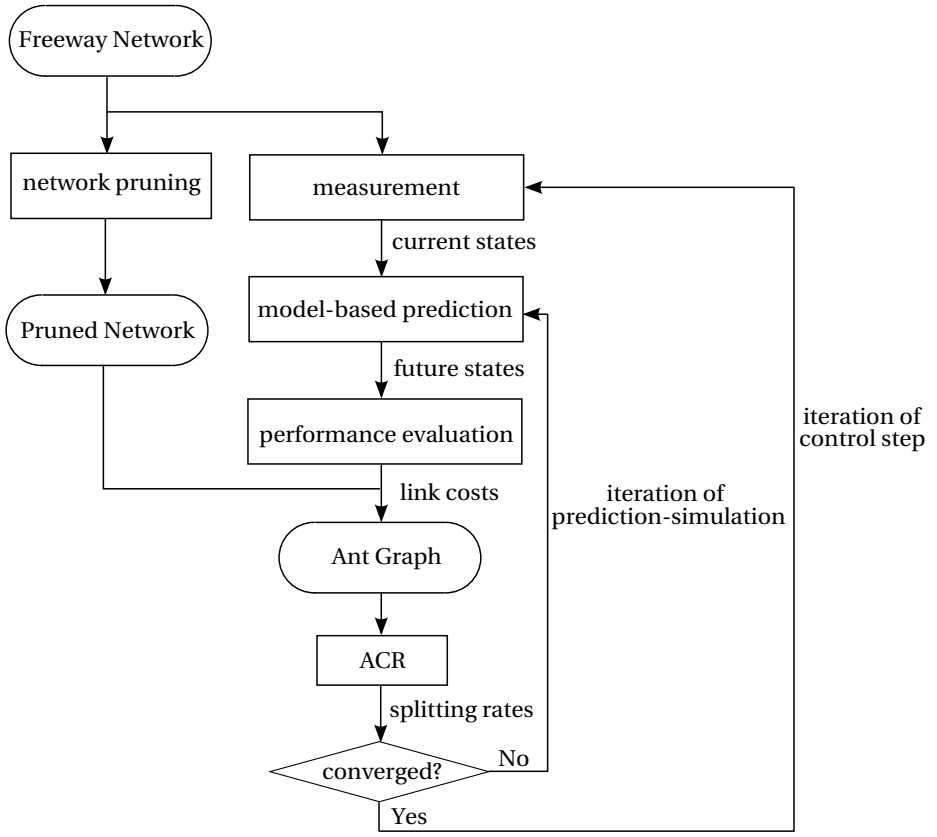
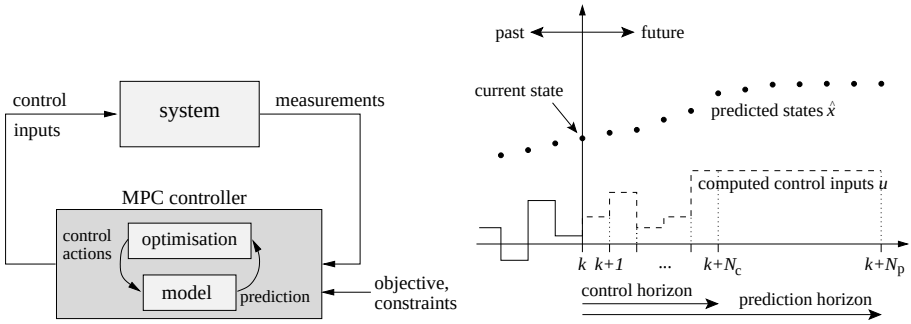


Figure 3.1: Structure of the proposed control strategy for dynamic traffic routing.

Therefore, such "unnecessary" links and nodes should be removed to reduce the complexity of the traffic network, so as to decrease the computational burden of the ACR algorithm. It must be emphasized that the network pruning step involves a static optimization problem, which is not solved by an ACO-based algorithm but by linear programming, and which is only performed once. For on-line dynamic traffic routing, the MPC controller is repeatedly executed using ACO-SP as the optimization method. More specifically, at each control step, the current state of the traffic network is measured, and the future state is predicted by a dynamic traffic model for a certain prediction period. Then, both the current and future traffic states are used to calculate the link cost, which will be assigned to each arc in the ant graph. The ACO-SP algorithm solved the dynamic traffic routing problem based on this ant graph, and as a result, the control signals in the traffic network — splitting rates, are determined according to the assignment result of ants in the ant graph. Such a prediction-optimization loop is repeated at each control step until a given convergence criterion is satisfied, and then the resulting splitting rate is applied to the real traffic network. Next, both the control horizon and the prediction horizon of the MPC controller is shifted one sample time step forward, and the whole process is repeated. The structure of the proposed control strategy is illustrated in Figure 3.1.

The rest of this chapter is structured as follows. Section 3.2 briefly reviews the MPC ap-



(a) MPC control loop (b) Prediction and control horizon in MPC

Figure 3.2: Schematic representation of the MPC approach.

Table 3.1: Symbols used for MPC

symbol	meaning
k	discrete time step for the simulation process
k_c	discrete time step for the controller
T	length of simulation sample time
T_c	length of control sample time
H_p	prediction horizon
H_c	control horizon
$x(k)$	process state
$\hat{\mathbf{x}}(k)$	vector of the predicted states based on the knowledge at simulation step k
$u(k_c)$	control variable
$\mathbf{u}(k_c)$	vector of the control variables based on the knowledge at control step k_c
$f(\cdot)$	process state update function
$J(\cdot)$	objective function
$\mathbf{u}^*(k_c)$	vector of the optimal control variables based on the knowledge at control step k_c

proach. Next, the extended version of the ACO-SP algorithm is introduced in Section 3.3. In Section 3.4, a two-step control strategy — network pruning and the ACR algorithm — is elaborated on. Section 3.5 illustrates the proposed approach using a study case involving the freeway network in the Walcheren area in the Netherlands. A short discussion finally concludes the chapter in Section 3.6.

3.2 Model Predictive Control

Model Predictive Control (MPC) [95, 118] is a methodology that implements and repeatedly applies optimal control in a rolling horizon way. The symbols used for MPC are listed in Table 3.1. The MPC approach makes an explicit difference between the simulation time step k for the process model and the control time step k_c for the controller. The relationship between the length T of simulation sample time and the length T_c of control sample time is $T_c = MT$, with M an integer.

A general MPC process is shown in Figure 3.2(a), which can be described by three major parts:

- **Prediction**: The prediction is made at each control step k_c, for which the corresponding simulation time step is $k = Mk_c$. The process is described by the discrete-time state update function:

$$x(k+1) = f(x(k), u(k_c)), \text{ for } k = Mk_c, Mk_c+1, \ldots, M(k_c+1)-1 \tag{3.1}$$

  with $f(\cdot)$ the state update function, $x(k)$ the state of the process at simulation step k, and $u(k_c)$ the control variables at control step k_c. The prediction is made by repeatedly applying (3.1) for the simulation period $[kT, (k+MH_p)T)$, with H_p the length of prediction horizon in units of control steps. The inputs for the model-based prediction are the current process state $x(k)$, and the control vector $\mathbf{u}(k_c) = [u^T(k_c|k_c)\ u^T(k_c+1|k_c)\ \ldots\ u^T(k_c+H_p-1|k_c)]^T$, with $u(\cdot|k_c)$ the control variables for the corresponding control step based on information available at control step k_c. Based on these inputs, the future evolution of the process is predicted, and is denoted by a vector $\hat{\mathbf{x}}(k) = [\hat{x}^T(k+1|k)\ \hat{x}^T(k+2|k)\ \ldots\ \hat{x}^T(k+MH_p-1|k)]^T$.

- **Optimization**: An optimization algorithm is applied to find the optimal control vector $\mathbf{u}^*(k_c)$ that minimizes the objective function $J(\hat{\mathbf{x}}(k), \mathbf{u}(k_c))$ during the prediction period $[kT, (k+MH_p)T)$. The control variables are only optimized for the so-called control horizon $H_c (\leqslant H_p)$. If H_c is smaller than H_p, the control variables follows the constraint $u(k_c+k_c'|k_c) = u(k_c+H_c-1|k_c)$ for $k_c' = H_c, H_c+1, \ldots, H_p-1$.

- **Control action**: Only the first sample $u^*(k_c|k_c)$ of the optimal control vector $\mathbf{u}^*(k_c)$ is applied to the process. Then, the procedure from prediction to control action is repeated at control step k_c+1 with the prediction horizon shifted one time step ahead, and so on (see Figure 3.2(b)).

3.3 Extension of ACO-SP for Dynamic Traffic Routing

The original ACO-SP algorithm is now extended for the dynamic traffic routing problems using the following elements: colored ants, constraints on the number of vehicles, and dynamic link cost.

3.3.1 Colored Ants

Colored ants are used to distinguish vehicles with different destinations. The vehicles with destination d correspond to ants with color η_d. Colored ants only move and deposit the pheromone in an ant graph with the corresponding color. For each ant graph, the number $N_{\text{ant,total},d}$ of ants with color η_d depends on the size of the corresponding pruned network (see Section 3.4.1 for more details). The more nodes and links the pruned network has, the more ants are needed to get a good performance. The number $N_{\text{ant,total},d}$ of ants is assumed to be fixed and time-invariant during each ACR run. Therefore, a dynamic factor $\mu_d(k_c)$ is defined to relate the number $N_{\text{ant,total},d}$ of ants to the number $N_{\text{veh,in},d}(k_c)$ of incoming

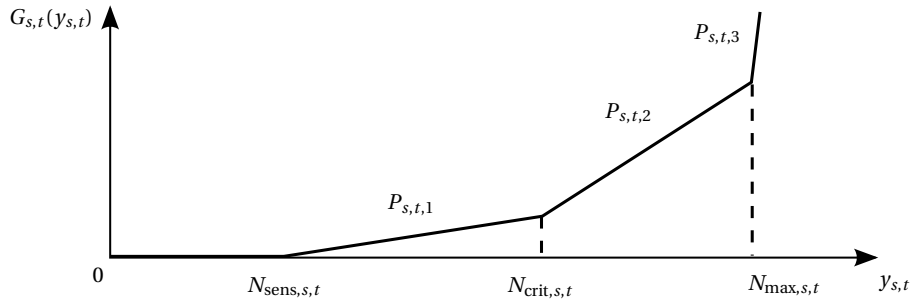


Figure 3.3: Piecewise affine stench functions $G_{s,t}$.

vehicles with destination d at each control step k_c:

$$\mu_d(k_c) = \frac{N_{\text{ant,total},d}}{N_{\text{veh,in},d}(k_c)} \,. \tag{3.2}$$

This factor indicates how many vehicles an ant with color η_d represents at each control step k_c. Note that the stench pheromone is not colored because it is used to push all of the ants away from an arc.

3.3.2 Constraints on the Number of Vehicles

Several constraints on the number of vehicles are considered for each link in the traffic network. More specifically, there is a number corresponding to the link capacity that can never be exceeded, a number corresponding to critical traffic density that should preferably not be exceeded since otherwise traffic congestion may occur, and probably one or more threshold numbers for sensitive zones where a limit portion of vehicles are allowed to pass for societal or environmental reasons, e.g., hospitals and schools. All these constraints and thresholds are implemented through the stench pheromone function $G_{s,t}(\cdot)$ defined on each arc (s,t) in the ant graph. For instance, the function $G_{s,t}$ can be specified as a piecewise affine function as (see Figure 3.3):

$$G_{s,t}(y_{s,t}) = \begin{cases} 0\,, & \text{if } 0 \leqslant y_{s,t} < N_{\text{sens},s,t} \\ P_{s,t,1}(y_{s,t} - N_{\text{sens},s,t})\,, & \text{if } N_{\text{sens},s,t} \leqslant y_{s,t} < N_{\text{crit},s,t} \\ P_{s,t,2}(y_{s,t} - N_{\text{crit},s,t}) + B_{s,t,1}\,, & \text{if } N_{\text{crit},s,t} \leqslant y_{s,t} < N_{\text{max},s,t} \\ P_{s,t,3}(y_{s,t} - N_{\text{max},s,t}) + B_{s,t,2}\,, & \text{if } y_{s,t} \geqslant N_{\text{max},s,t} \end{cases} \tag{3.3}$$

with $N_{\text{sens},s,t}$ the number of ants corresponding to the threshold number of incoming vehicles on link $\ell(s,t)$ in sensitive zones, $N_{\text{crit},s,t}$ the number of ants corresponding to the critical traffic density on link $\ell(s,t)$, $N_{\text{max},s,t}$ the number of ants corresponding to the capacity on link $\ell(s,t)$ in the traffic network, $P_{s,t,1}$, $P_{s,t,2}$, $P_{s,t,3}$ the slopes of the affine functions in each piece of the function $G_{s,t}(\cdot)$, with a relationship $P_{s,t,1} < P_{s,t,2} < P_{s,t,3}$, and $B_{s,t,1} = P_{s,t,1}(N_{\text{crit},s,t} - N_{\text{sens},s,t})$ and $B_{s,t,2} = P_{s,t,2}(N_{\text{max},s,t} - N_{\text{crit},s,t}) + P_{s,t,1}(N_{\text{crit},s,t} - N_{\text{sens},s,t})$ the constants to guarantee continuity of the function $G_{s,t}(\cdot)$.

Recall that the relationship between the number $N_{\text{ant,total},d}$ of ants with color η_d and the

number of vehicles $N_{\text{veh,in},d}(k_c)$ with destination d at control step k_c is characterized by the factor $\mu_d(k_c)$ (cf. (3.2)). Instead of directly adding up the numbers of ants for different colors, the number $y_{s,t}$ of ants on each arc (s,t) is calculated using so-called *standard* ants, which have a one-to-one relationship with the vehicles:

$$y_{s,t}(k_c) = \sum_{d \in \mathscr{D}} \frac{y_{s,t,d}(k_c)}{\mu_d(k_c)}. \tag{3.4}$$

Note that the numbers $N_{\text{sens},s,t}$, $N_{\text{crit},s,t}$, and $N_{\text{max},s,t}$ also represent standard ants.

Remark 3.1 In practice, the values of the slopes $P_{s,t,1}$, $P_{s,t,2}$, and $P_{s,t,3}$ can be dynamically assigned to better serve and manage the traffic network. During a day, the values of the slopes can be increased to allow fewer vehicles to enter the sensitive zones so as to avoid danger, pollution, and noise, or the values of the slopes can be decreased to allow more vehicles to enter the sensitive zones to guarantee the smoothness of traffic flow and to reduce the burden for other zones. □

3.3.3 Dynamic link cost

A dynamic traffic model is used to assign a dynamic cost to each link in the traffic network, and in this chapter, the METANET model (see more details in Appendix B) is chosen. The link cost $\varphi_m(k_c)$ is calculated based on the traffic state at each control step k_c. In this chapter, only the total time spent (TTS) is considered as the link cost. Two different methods to calculate $\varphi_m(k_c)$ are presented, called quasi-static and fully-dynamic, respectively. The main difference between the two cases is whether or not the cost $\varphi_m(k_c)$ is being updated while ACR is running.

Quasi-static case

The quasi-static case first uses the predicted traffic state, in particular the speed $v_{m,i}(k_c)$ in this chapter, to calculate the travel time on segment i of link m, for all simulation steps $k = Mk_c, Mk_c+1, \ldots, M(k_c+H_p)-1$ in the prediction period, and then sums up the travel time on all the segments of link m for all the simulation steps to calculate $\varphi_m(k_c)$, which is formulated as:

$$\varphi_m(k_c) = \sum_{k=Mk_c}^{M(k_c+H_p)-1} \sum_{i=1}^{N_m} \frac{L_m}{v_{m,i}(k)}, \tag{3.5}$$

with L_m the length of each segment, and N_m the number of segments on link m. As a matter of fact, the cost $\varphi_m(k_c)$ is kept fixed at each control step k_c when ACR is running. At the end of each iteration of ACR, the speed $v_{m,i}(k_c)$ is updated based on the new control variable (cf. (3.15)), such that the new link cost is obtained. Then, a new iteration of ACR is repeated.

Fully-dynamic case In the fully-dynamic case, the cost $\varphi_m(k_c)$ varies while ACR is running. Note that in principle ants move on the ant graph, which is based on the (pruned) traffic network. However, for the sake of the simplicity of the explanation, in this section we assume that ants travel on the links of the traffic network. Therefore, for a given link m, each ant a traveling on it will incur a possibly different cost $\varphi_{m,a}(k_c)$. Figure 3.4 describes how to track

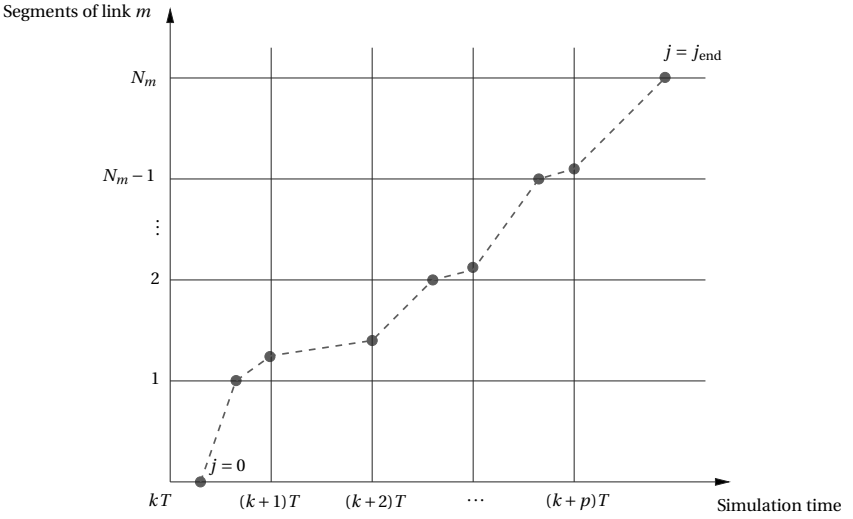


Figure 3.4: Travel time determined by ant a on link m.

the travel time of ant a according to the dynamic speed profile $v_{m,i}(k)$ for all link-segment pairs (m,i), and for the simulation time steps k. Since the speed of the traffic model varies in different segments and at different time steps, the point where the speed $v_{m,i}(k)$ changes to a new speed $v_{m,i}(k+1)$ or $v_{m,i+1}(k)$ is called a *jump point* (indicated by j). Obviously, jump points can only placed on the boundary of segments or on the simulation time steps. If a jump point j is placed in the segment i, the segment i is denoted as i_j to illustrate the relationship between jump point j and segment i. Similarly, if a jump point j is placed at the simulation step k, the simulation step k is denoted as k_j.

At a given jump point j, ant a predicts the travel time between jump points j and $j+1$, and then jumps to $j+1$ to make a new prediction. Ants will keep this *predict-jump-predict* mode until they reach the end of the link m, which yields the last jump point j_{end}. The time instant when ant a on link m is at jump point j is denoted as $t_{m,a,j}(k_c)$. Figure 3.5 illustrates the four possible transitions of ant a from jump points j to $j+1$. For each case, the time instant $t_{m,a,j+1}(k_c)$ is calculated as follows:

- Case A: $t_{m,a,j+1}(k_c) = t_{m,a,j}(k_c) + \dfrac{L_m - D_{m,a,j}(k_c)}{v_{m,i_j}(k_j)}$
- Case B: $t_{m,a,j+1}(k_c) = (k_{j+1}+1)\,T$
- Case C: $t_{m,a,j+1}(k_c) = t_{m,a,j}(k_c) + \dfrac{L_m}{v_{m,i_j}(k_j)}$
- Case D: $t_{m,a,j+1}(k_c) = (k_{j+1}+1)\,T$

where $D_{m,a,j}(k_c)$ is the distance between the beginning of segment i_j and the jump point j as shown in Figure 3.5. The distance between jump points j and $j+1$ is denoted as $d_{m,a,j}$,

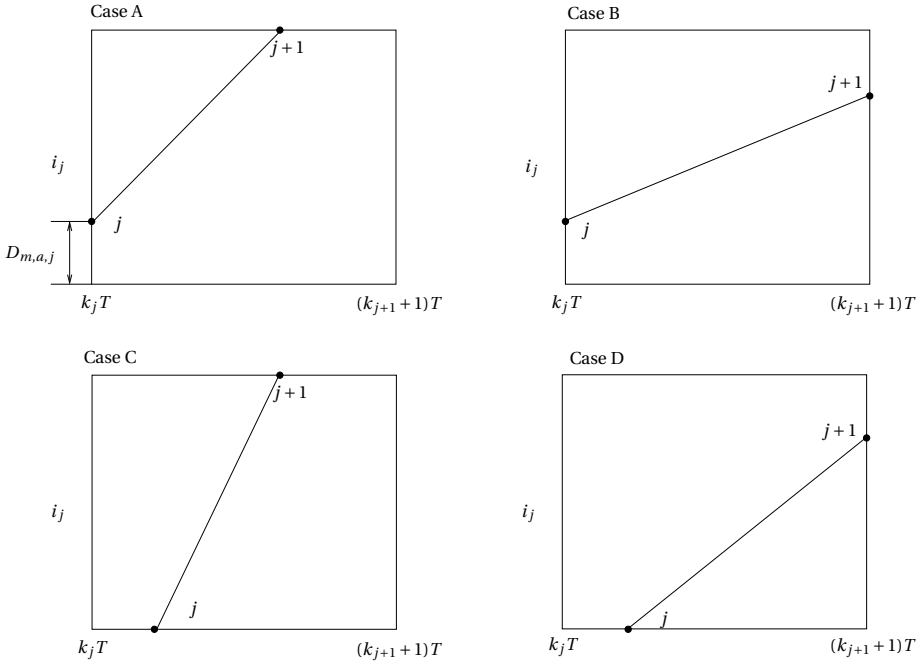


Figure 3.5: Four possible ways in which ant a on link m can jump from point j to point $j+1$ between the time intervals kT and $(k+1)T$.

which is expressed as

$$d_{m,a,j}(k_c) = v_{m,i_j}\left(k_j\right)\left(t_{m,a,j+1}(k_c) - t_{m,a,j}(k_c)\right). \tag{3.6}$$

Therefore, the total distance between the jump point $j=0$ and current jump point j is the sum $\sum_{z=0}^{j-1} d_{m,a,z}$. Besides, the length of each segment of link m equals L_m. The distance $D_{m,a,j}(k_c)$ can then be obtained by:

$$D_{m,a,j}(k_c) = \mathrm{mod}\left(\sum_{z=0}^{j-1} d_{m,a,z}(k_c),\, L_m\right), \tag{3.7}$$

where mod denotes the modulo operator. Note that Case C can be considered as $D_{m,a,j}(k_c) = 0$, so the equation of Case A also holds for Case C. Hence, a unified formulation of the time instant $t_{m,a,j}(k_c)$ for all four cases is:

$$t_{m,a,j+1}(k_c) = \min\left(t_{m,a,j}(k_c) + \frac{L_m - D_{m,a,j}(k_c)}{v_{m,i_j}\left(k_j\right)}, \left(k_{j+1}+1\right)T\right). \tag{3.8}$$

The time instant $t_{m,a,0}(k_c)$ is recorded when ant a enters link m, and the time instant $t_{m,a,j_{\mathrm{end}}}(k_c)$ of the last jump point j_{end} is calculated by using (3.8). The cost $\varphi_{m,a}(k_c)$ on link m is then the travel time of ant a:

$$\varphi_{m,a}(k_c) = t_{m,a,j_{\mathrm{end}}}(k_c) - t_{m,a,0}(k_c). \tag{3.9}$$

3.4 Control Strategy

3.4.1 Network Pruning

The network pruning step aims at removing the "unnecessary" links and nodes in a traffic network, i.e. those links and nodes that only belong to routes that are relatively long and that will therefore not be favored by drivers. This step first runs for each OD-pair to yield the K best routes to set up a pruned network, with K an integer that could be different for each OD-pair. Next, the pruned networks with the same destination d are merged, and then mapped into an ant subgraph with color η_d. The number of the ant subgraphs equals the number N_d of the destinations of the traffic network. At last, all the ant subgraphs are combined as one overall ant graph, used for ACO-SP. Recall that the colored ants only move and deposit the pheromone in the ant subgraph with the corresponding color, but the stench pheromone is calculated by the total number of ants on each arc, thus it is put down in the overall ant graph.

Two different approaches are considered for network pruning, which are the K-shortest routes method, and the linear programming method.

- The simplest way is to choose the K shortest loopless routes to construct the pruned network. Several algorithms have been developed for this, e.g., [68, 79, 146]. These algorithms can be used to determine the K shortest routes for each OD-pair by considering the length of each link, or the average travel time based on historical data. Afterwards, the links that do not belong to any route are removed, and, as a result, the pruned network is obtained. However, this method cannot guarantee that the capacity of each link on the routes is never exceeded.

- One way to address the capacity issue is to use linear programming with capacity constraints to find the links with the highest link flows to determine the K best routes for each OD-pair. This method involves a quasi-static approach by considering the traffic situations in a representative day. More specifically, the representative day is divided into several time slots (e.g., the morning rush hour, the non-busy midday period, and the evening rush hour), and for each time slot, the traffic flows $q_{m,d}$ on each link m will be determined for each destination d such that the total travel time is minimized subject to capacity constraints. This is done as follows. For each link m, the average travel time is defined as $t_m = L_m / v_m$ with L_m the length of link m, and v_m the average speed on link m. The linear programming problem for minimizing the total travel time is defined as:

$$\min_{q_{m,d}} \sum_{d\in\mathscr{D}} \sum_{m\in\mathscr{M}} T \cdot t_m \cdot q_{m,d} \tag{3.10}$$

subject to

$$q_{m,d} = q_{\text{in},o,d}, \quad \forall (o,d) \in \mathscr{H} \times \mathscr{D}, \forall m \in \mathscr{O}_o \tag{3.11}$$

$$\sum_{d\in\mathscr{D}} \sum_{m\in\mathscr{I}_n} q_{m,d} = \sum_{d\in\mathscr{D}} \sum_{m\in\mathscr{O}_n} q_{m,d}, \quad \forall n \in \mathscr{N} \tag{3.12}$$

$$\sum_{d\in\mathscr{D}} q_{m,d} \leqslant q_{\max,m}, \quad \forall m \in \mathscr{M} \tag{3.13}$$

$$q_{m,d} \geqslant 0, \quad \forall m \in \mathscr{M}, \forall d \in \mathscr{D}, \tag{3.14}$$

with T the length of the simulation sample time, $q_{\max,m}$ the capacity of link m, $\mathcal{O}_o$ the set of outgoing links of origin o, and $\mathcal{I}_n$ and $\mathcal{O}_n$ respectively the sets of incoming and outgoing links of node n. Note that (3.10) minimizes the total travel time, because $T \cdot q_{m,d}$ expresses the number of vehicles on link m per simulation step, thus $T \cdot t_m \cdot q_{m,d}$ corresponds to the total travel time on link m. Moreover, (3.12) states that the inflow of node n equals the outflow of node n (conservation of vehicles). It is easy to verify that (3.10)–(3.14) is a linear programming problem, which can be solved very efficiently using, e.g., a simplex method or an interior-point algorithm [103, 142]. Once the solution is determined, only the links satisfying the condition $\sum_{d \in \mathcal{D}} q_{m,d} \geqslant q_{\text{thresh},m}$, with $q_{\text{thresh},m}$ a pre-defined positive threshold value, are selected, and the other links are removed. Next, the links that do not belong to any route for the OD-pair are also removed. The remaining links are kept as the pruned network. However, the shortcoming of this method is that possibly no route from origin o to destination d is contained in the set of the selected links. If this occurs, the value $q_{\text{thresh},m}$ should be decreased to guarantee at least one route.

Remark 3.2 One can combine the two methods introduced above to address both the capacity constraint issue of the K shortest routes method and the no-route problem of the linear programming method. First, the K shortest routes are determined in order to construct a pruned network; next, the linear programming problem (3.10)–(3.14) is solved for the pruned network. If the linear programming problem is feasible, the capacity problem does not occur; otherwise, the value K is augmented, and the procedure is repeated until a feasible linear programming problem is yielded. □

Sometimes, when using a K-shortest-routes algorithm, the problem may occur that there are too many overlapping links in the resulting routes. In that way, the pruned network does not have sufficient links to build up an ant graph. If that happens, besides simply increasing the value of K to find some more candidate routes, one could use a dedicated K-shortest-routes algorithm that avoids having too many links in common between different routes (see e.g. [148]), or disjoint shortest path algorithm (see e.g. [130]).

3.4.2 ACR run

The MPC control scheme is shown in Figure 3.6. The traffic model predicts the evolution of the traffic network at every simulation step k, while the ACR algorithm only runs at each control step k_c. We start the prediction of the traffic system at control step k_c, for which the corresponding simulation step is $k = Mk_c$. The process is described by the selected dynamic traffic model during the prediction period $[k_c T_c, (k_c + H_p) T_c)$. The prediction requires three inputs:

- the current traffic states[3] $x_{m,i}(k)$ for each segment i on link m, and $x_o(k)$ for each origin o at simulation time step $k = Mk_c$;
- a vector of the expected inflows of the network at each simulation step during the prediction period: $\hat{\mathbf{q}}_{\text{in},o,d}(k) = [q_{\text{in},o,d}(k)\ q_{\text{in},o,d}(k+1)\ \cdots\ q_{\text{in},o,d}(k+MH_p-1)]^T$ for all OD-pairs;

[3] The state can be the number of vehicles, flow, speed, density, emission, etc, according to different traffic models.

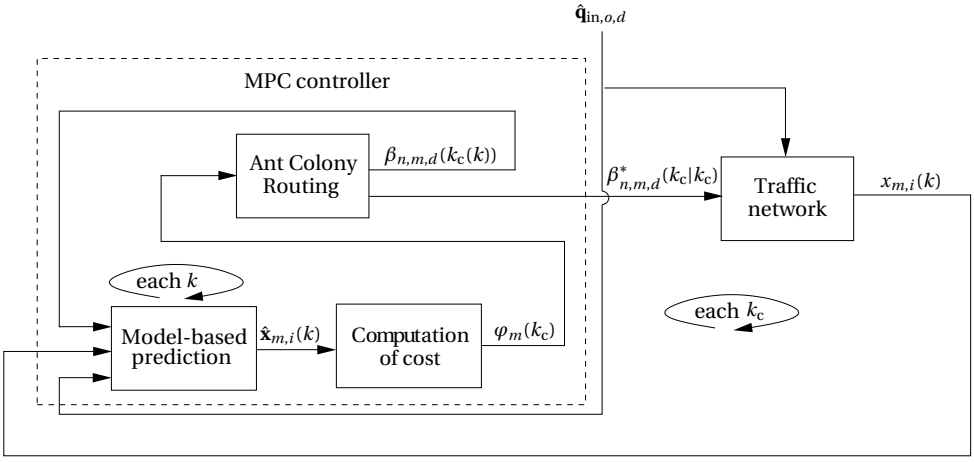


Figure 3.6: Closed-loop control of a traffic system with the ACR algorithm and the dynamic traffic prediction model.

- the currently imposed control signal — splitting rates $\beta_{n,m,d}(k_c(k))$ for node n, outgoing link m, and destination d, at control step $k_c = k_c(k) = \lfloor k/M \rfloor$, with $k_c(k)$ the value of the control step counter k_c corresponding to simulation step k, and the operator $\lfloor \cdot \rfloor$ the largest integer less than or equal to the function argument.

The output of model-based prediction is the future traffic state vector $\hat{\mathbf{x}}_{m,i}(k) = [\hat{x}_{m,i}(k+1|k)\ \hat{x}_{m,i}(k+2|k)\ \cdots\ \hat{x}_{m,i}k+MH_p-1|k]^T$, with $\hat{x}_{m,i}(\cdot|k)$ the predicted state at the corresponding simulation step based on knowledge at simulation step k.

The ACR algorithm is used as the optimization method in the MPC controller, aiming at generating the splitting rates of the traffic flow to each destination $d \in \mathscr{D}$ at each node $n \in \mathscr{N}$ in the traffic network. After the ACR algorithm has terminated, the number of ants $y_{s,t,d}$ of color η_d that have traveled on each link (s,t) is determined. These values are used to update the splitting rates for each $n \in \mathscr{N}$, $m \in \mathscr{O}_n$, and each $d \in \mathscr{D}$, :

$$\beta_{n,m,d}(k_c(k)) = \frac{y_{s,t,d}(k_c)}{\sum_{t' \in \mathscr{S}_{s,d}} y_{s,t',d}(k_c)} \text{ , with } m = \ell(s,t) \tag{3.15}$$

with $\mathscr{S}_{s,d}$ the set of the nodes in the ant graph that are connected to node s for color η_d. Moreover, if a link m in the traffic network does not have a corresponding arc in the ant graph for destination d, we set $\beta_{n,m,d}(k_c(k)) = 0$. All of the resulting splitting rates will be applied to the prediction model, and as a result, we will obtain new traffic states for updating the cost $\varphi_m(k_c)$, as well as new values of $y_{s,t,d}(k_c)$. We repeatedly run the prediction-ACR updating process until one of the criteria below (or their combination) is satisfied:

1. The maximum number of iteration steps N_{fixed} is reached;

2. $|\Delta\beta_{n,m,d}(k_c(k))| < \epsilon_\beta$, for all n, m, and d, where $\Delta\beta_{n,m,d}(k_c(k))$ is the difference between the new and the previous value of $\beta_{n,m,d}(k_c(k))$, and $\epsilon_\beta > 0$ is a predefined tolerance.

Figure 3.7: Map of the Walcheren area (source: google maps).

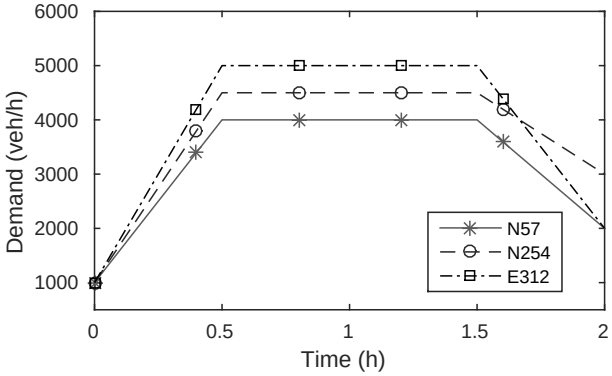
Then the final splitting rates are the outputs of the MPC controller, and they are used to control the real traffic system via lower level controllers, such as dynamic matrix panels with route information, on-board route guidance devices, and so on. Moreover, a financial compensation or penalty can be applied to convince drivers to follow the suggested route guidance instructions for the system sake. For instance, drivers have to pay if they travel in so-called congestion charge zones, while they need not pay if they travel outside of these areas [123], (see e.g. congestion charge schemes in London, Stockholm and Singapore). In the Netherlands, the government is testing a reward system called Spitsmijden [15] that tracks commuters and pays a range of rewards (€3, €5, and €7 per day) to those who avoided traveling during the morning peak (7:30 am to 9:30 am). More information about the implementation of the route guidance mechanism can be found in e.g. [14].

3.5 Case Study

The ACR algorithm is now tested in a simulation of the Walcheren area in Zeeland, the Netherlands [76]. First, the set-up of the case study and the routing scenario are described in Section 3.5.1. Section 3.5.2 shows the result of the pruning step, which consists in finding a reduced network for each destination. To analyze the ACR algorithm, we first compare a congested traffic situation with and without ACR control in Section 3.5.3, and then compare the ACR algorithm with two other dynamic routing methods in Section 3.5.4.

3.5.1 Simulation Scenario

A map of the Walcheren area is shown in Figure 3.7. Middelburg at its center is the provincial capital and the biggest municipality in the area. Vlissingen in the south is the main harbor



	d_1	d_2	d_3
o_1	30%	30%	40%
o_2	50%	30%	20%
o_3	20%	40%	40%

Figure 3.8: Inflows of the freeway network from the different origins, and the fractions of each inflow traveling to different destinations.

and the second biggest municipality. The third biggest municipality is Veere in the northeast. Moreover, the whole area is connected to the mainland by three main freeways, which are the N57 in the north, the E312 in the middle, and the N254 in the south. The entire freeway network has 142 links and 62 nodes, including the origins and the destinations.

We consider a scenario that drivers outside Walcheren enter the area only through the N57, the E312, or the N254, and they are going to Middelburg, Vlissingen, or Veere by using the freeway network only. The origins of the traffic network are put at the entrances of the N57 (o_1), the E312 (o_2), and the N254 (o_3), indicated by the large white dots in Figure 3.7. The destination nodes are put at the exits of the freeway network in Middelburg (d_1), Veere (d_2), and Vlissingen (d_3), indicated by the large red, blue, and green dots respectively. The sensitive zones in this case study include the links across the urban areas, where a threshold value for the traffic density is set as $\rho_{\text{sens},m} = 20$ veh/km/lane. The simulation period is set to 2 hours, with an empty network as the initial state. The inflow from each origin, with the fractions of each inflow traveling to different destinations, is shown in Figure 3.8.

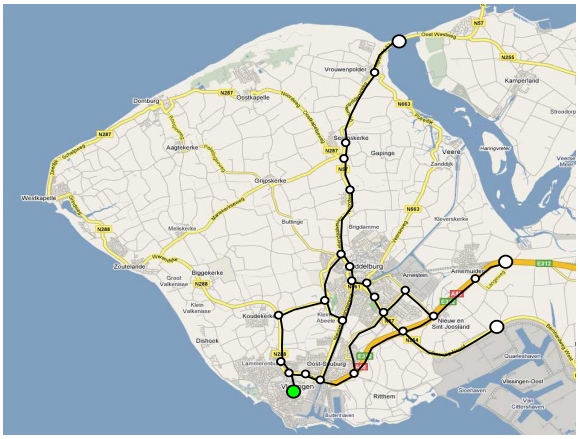
For this scenario, the routing instructions are optimized using the ACR approach proposed in this chapter. Although we have discussed two different ways to compute the link cost in the network, and the fully-dynamic case is much more accurate than the quasi-static case, we only implement the quasi-static case in this case study. This is because the fully-dynamic case calculates the link cost for each jump point, which results in a very high computational burden. On the contrary, in the quasi-static case we only need to calculate the average travel time on each link, and in that way we can achieve a balanced trade-off between computation speed and accuracy.

3.5.2 Network Pruning

We first use Yen's K-shortest-loopless routes algorithm [146] to find the K shortest routes for each OD pair, with $K = 3$. Then, we solve the linear programming problem (3.10)–(3.14) to verify whether the flow on any link of these routes exceeds the link capacity. Since the result shows that the link capacity is not exceeded, the value of K should not to be augmented in this case.

(a) Pruned network for destination d_1

(b) Pruned network for destination d_2



(c) Pruned network for destination d_3

Figure 3.9: Pruned networks for different destinations.

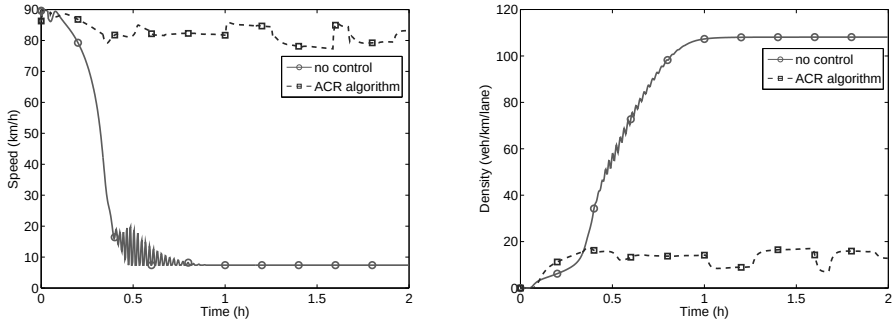


Figure 3.10: Congested traffic conditions with and without ACR control on the representative link.

Three different pruned networks for different destinations are established, as shown in Figure 3.9. The first pruned network has 26 links and 21 nodes, the second pruned network has 21 links and 19 nodes, and the third pruned network has 30 links and 26 nodes.

3.5.3 Routing Results for ACR

In this case study, we assume that the resulting routing instructions will be perfectly followed by the drivers. The reason for this is that the primary objective of this case study is to discuss the functioning of the ACR algorithm. By assuming full compliance, the functioning can be more easily compared with other methods.

We compare the traffic conditions with and without ACR control. Figure 3.10 shows an example of the traffic density and travel speed on a representative link, indicated by "RepLink" in Figure 3.7. This representative link has been chosen because it belongs to all three subnetworks. Moreover, this link has different statuses in different subnetworks:

1. It belongs to the shortest route from origin o_1 to destination d_3 in the green subnetwork, and it is shared by all the routes from o_1 to d_3;

2. It belongs to the shortest route from origin o_1 to destination d_1 in the red subnetwork, but there is an alternative route that does not include the representative link;

3. It does not belong to the shortest route from origin o_1 to destination d_2 in the blue subnetwork.

When there is no control, we assume that all the drivers always choose the shortest-distance route. As shown in Figure 3.10, the maximal density on the link is eventually reached, resulting in an extremely low travel speed. However, if the ACR algorithm is applied, no traffic congestion will occur.

3.5.4 Comparison with Other Methods

To evaluate the performance and the computational efficiency of the ACR algorithm, we compare our approach with two other dynamic traffic routing methods. The first one is the non-linear optimal control approach of [87], which has been used in the case study in

Chapter 2, and the second one is the time-dependent shortest routes method described in [136].

The non-linear optimal control problem of [87] is formulated as:

$$\begin{aligned} & \min\ J = \min\ J_{\text{TTS}} + J_{\text{pen}} \\ \text{subject to}\ & x(k+1) = f(x(k), \beta(k_c(k)), d(k)), \\ & 0 \leqslant \beta(k_c(k)) \leqslant 1, \\ & \text{for}\ \ k = 1, 2, \ldots, K, \end{aligned} \tag{3.16}$$

with $f(\cdot)$ the traffic model, $x(k)$ the traffic state at simulation step k, $d(k)$ the demand at simulation step k, $\beta(k_c(k))$ the splitting rate used as the control variable at the control step $k_c(k)$, and K the simulation horizon. The sub-objective functions J_{TTS} and J_{pen} are calculated as:

$$\begin{aligned} J_{\text{TTS}} &= T \cdot \sum_{k=1}^{K} \sum_{m \in \mathscr{M}} \sum_{i=1}^{N_m} \rho_{m,i}(k) \frac{L_m \lambda_m}{N_m} \\ J_{\text{pen}} &= P \cdot \sum_{k=1}^{K} \sum_{m \in \mathscr{M}} \sum_{i=1}^{N_m} \left(\rho_{m,i}(k) - \rho_m^{\text{thresh}} \right) \frac{L_m \lambda_m}{N_m} \end{aligned} \tag{3.17}$$

with T the length of the simulation time step, P a penalty factor, $\rho_{m,i}(k)$ the density on segment i of link m at simulation time step k, ρ_m^{thresh} the threshold density on link m, L_m the length of link m, λ_m the number of lanes of link m, $\mathscr{M}$ the set of all the links in the network, and N_m the number of segments on the link m. Note that unlike in Chapter 2, the objective function J is not exactly the same as the one formulated in [87]. The paper [87] only aims at minimizing the TTS (J_{TTS}), while in this chapter we want to find a balance between minimizing the TTS and controlling the numbers of vehicles on the links. Therefore, we add a term J_{pen} in the objective function. The sub-objective function J_{pen} represents a soft constraint, and works in a similar way as the stench pheromone function in the ACR algorithm, by penalizing the number of vehicles exceeding a threshold number on each link. The value of $\rho_{\text{thresh},m}$ is equal to either the critical density $\rho_{\text{crit},m} = 33.5$ veh/km/lane in non-sensitive zones, or to a pre-defined value $\rho_{\text{sens},m} = 20$ veh/km/lane in sensitive zones. Moreover, the optimality conditions are expressed in terms of the discrete-time Hamilton function, fulfilled by the Karush-Kuhn-Tucker conditions, and a numerical gradient-based algorithm is used to solve the discrete-time optimal control problem (see [87] for more details).

The time-dependent shortest routes approach of [136] uses an iterative algorithm to incrementally distribute the traffic flow over more and more routes. Each iteration consists of three steps:

- Step 1: In the first iteration, all links get assigned the free-flow travel time; while in subsequent iterations, a simple traffic model based on the speed-density fundamental diagram is used to determine the travel times on links that carry traffic (other links still get assigned the free-flow travel time).
- Step 2: Dijkstra's shortest path algorithm is applied to search the shortest-time route from each origin to each destination.
- Step 3: In the first iteration, all vehicles are assigned to the shortest-time route; while in subsequent iterations, a part of the traffic from previously selected routes is redistributed to the new shortest-time route.

Table 3.2: Comparison of the results obtained with different algorithms for one MPC step. ACR stands for Ant Colony Routing, NOC stands for Nonlinear Optimal Control, and TDSP stands for Time-Dependent Shortest Paths

Methods	J_{TTS} [veh· h]	J_{pen} [veh]	Computation time [s]
ACR	3.8316×10^4	0	5.11×10^4
NOC	3.8318×10^4	7.903	2.31×10^5
TDSP	4.7116×10^4	6.303×10^2	2.86×10^2

In this way, the newly generated route always has a shorter travel time than the previously generated ones in the current iteration. Note that this approach does not have an explicit objective function J as in (3.16). In order to compare the three approaches, we define assessment functions J_{TTS} and J_{pen} in the same manner as in (3.17).

The comparison of results for one MPC step is shown in Table 3.2. It shows that in this case study the performance of the ACR algorithm is better than the other two approaches, since the ACR algorithm can achieve a more balanced trade-off between accuracy and computational efficiency, which is needed for on-line model-based traffic control. Compared with the non-linear optimal control (NOC) algorithm , we can see that the ACR algorithm can achieve a slightly better performance (3.8316×10^4 veh·h versus 3.8318×10^4 veh·h in J_{TTS}), satisfies the capacity constraints (0 veh versus 7.903 veh in J_{pen}), and moreover it requires one order of magnitude less computation time: 5.11×10^4 s versus 2.31×10^5 s. The fact that the values for J_{TTS} between ACR and non-linear optimal control are almost exactly the same is probably a coincidence, but it shows that ACR can yield almost the same performance as the established optimal control approach of [87], which requires much more computation time. Moreover, according to the nature of the ACO algorithm, artificial ants independently search the network in parallel. Therefore, if we have enough processors, we can further reduce the computation time.

In general, when solving the dynamic routing problem, one should consider a balance between required computation time and performance among these three methods. More specifically, if the computation time is more important than the performance, then the time-dependent shortest paths algorithm is recommended. Otherwise, ACR should be used since it requires less computation time than the non-linear optimal control.

3.6 Conclusions

We have proposed a novel Ant Colony Routing algorithm for solving the dynamic traffic routing problem. The ACR algorithm uses artificial ants to search in the ant network, and the resulting assignment of ants is used to determine the splitting rates in the traffic network. We have applied the ACR algorithm in a two-step control approach: network pruning and Model Predictive Control (MPC). Through removing some “unnecessary” links and routes, the network pruning part can reduce the size of the objective network such that ants can more efficiently search in the network. The MPC control part uses the novel ACR algorithm with the stench pheromone and colored ants to efficiently guide the vehicles from multiple origins to multiple destinations. A simulation-based case study involving the Walcheren area in the Netherlands has been performed. The results show that the ACR algorithm is suitable

for on-line optimization, and can achieve a well-balanced trade-off between control performance and computation speed. Moreover, from the implementation point of view, it is recommended to run the ACR algorithm using multi processors, for further improving the computational efficiency.

Part II

Chapter 4

Co-design of Freeway Network Topology and Control Measures

The two main directions to improve traffic flows in networks involve changing the network topology and introducing new traffic control measures. In this chapter, we consider a co-design approach to apply these two methods jointly to improve the interaction between different methods and to get a better overall performance. We aim at finding the optimal network topology and the optimal parameters of traffic control laws at the same time by solving a co-optimization problem. However, such an optimization problem is usually highly nonlinear and non-convex, and it possibly involves a mixed-integer form. Therefore, we discuss four different solution frameworks that can be used for solving the co-optimization problem, according to different requirements on the computational complexity and speed. A simulation-based study is implemented on the Singapore freeway network to illustrate the co-design approach and to compare the four different solution frameworks.

Research work of this chapter has been published in [37].

4.1 Introduction

In order to improve the performance of a traffic network, traffic authorities and policy makers usually pose this problem in one of the following two forms: changing the network topology or introducing traffic control measures. Network topology design involves construction works such as building new links or expanding existing links in the network. The advantage of this approach is that it may directly and effectively solve the capacity limitation problem, while the disadvantage is that the implementation may be very expensive and time-consuming, and sometimes the required free space may be not available. On the other hand, traffic control measures aim at a more efficient use of the existing infrastructure, without changing the network topology. However, this approach may be not able to improve traffic flows in some cases, e.g., when the total demand exceeds the capacity of the network. Therefore, we consider a co-design method that jointly optimizes the network topology design and the traffic control measures. Intuitively, a better performance of traffic networks is expected by doing co-design of the network topology and the control measures compared to optimizing each of them separately. In fact, we will show in this chapter that the co-design approach can indeed yield better results in terms of overall costs. Moreover, the co-design approach can be used to assist in the design of a new network, as it allows to compare different network topologies including variations in types and locations of traffic control measures. In

this chapter, we focus on freeway networks, but the co-design approach can also be easily adapted to urban traffic networks.

In a freeway network, topology design refers to adding or removing links, or to changing the numbers of lanes of links. It seems counterintuitive to remove links or lanes in order to improve the performance of the network, but in fact additional road capacity can sometimes induce extra traffic demand, and if not accurately predicted and planned, this extra traffic may lead to the road becoming congested sooner (a well-known example is the Braess paradox [23]). Moreover, from an environmental point of view, when freeway networks are built near or through existing communities, the quality of life in the neighborhoods is decreasing due to noise and pollution. In this case, it could be considered beneficial for societal reasons to remove links in the network. In summary, network topology design is a multifaceted problem, where different questions such as environmental impact, budgeting, safety, public inconvenience, etc., have to be considered together. Moreover, some post-design issues such as network maintenance after construction should also be taken into account.

Sometimes, it is not necessary to change the network topology in order to improve the performance of the network, because it is possible that the available infrastructure in the network is not effectively used. Traffic congestion can be caused by the fact that drivers choose routes selfishly or drive in an inappropriate manner. In this case, traffic authorities can introduce traffic control measures to influence driving behavior so that traffic congestion can be eliminated or at least reduced. Papageorgiou et al. [109] illustrate with a simple example that in a congested area, the total time spent (TTS) in the controlled case can be 14% less than in the uncontrolled case, if the traffic outflow is improved by 5% thanks to appropriate traffic control measures. However, this consequence also implies that any disturbance that reduces the traffic outflow with a few percents may significantly increase the TTS, and hence decrease the performance of the network.

While the network topology is not changed once determined for the given design period, the traffic control measures do need to be adapted to the time-varying traffic situations. Due to this different time scale, one usually chooses a standard "optimal" setting of the control strategy, which is static, when doing the topology design. However, this method is not accurate enough to capture the dynamic nature of the traffic flows in the network. Therefore, we introduce a so-called *parameterized traffic control* approach [147], where the parameters of the control laws are optimized according to a pre-defined objective function. The reason for using parameterized control is that for some other control approaches such as optimal control or model predictive control, the traffic control inputs usually consist of dynamic signals that vary on a minute to minute basis according to the time-varying traffic situations; however, in the parameterized control approach, the parameters of the control laws are considered fixed over the design period, and the control laws generate dynamic traffic signals based on the state of the traffic network. Moreover, we can even consider a more comprehensive quasi-dynamic setting (see Section 4.4.2), where the design period is divided into different sub-periods, and where each sub-period has a separate group of control law parameters. By using the predicted long-term future evolution of traffic demand, both the topology and the control law parameters can then be optimized jointly. It is however important to note that although we use the co-design approach to determine the parameters of control laws for the traffic control measures, this does not mean that these parameters should be fixed for the entire design period. Instead, we can still use online control measures, and regularly retune or optimize the parameters of the control law based on the real traffic situation, via e.g. standard traffic control strategies, optimal control, or model predictive control.

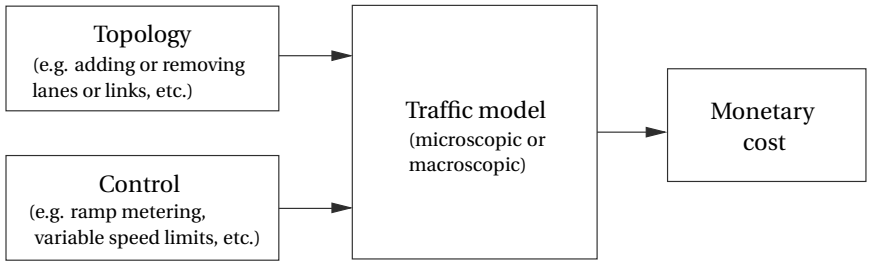


Figure 4.1: Representative scheme of the co-design problem.

The main aim of solving the co-design problem is to find the optimal topology design decisions and the optimal parameters of the control laws. In order to obtain those optimal solutions, both topology design decisions and traffic control measures are applied to a traffic model, and a cost is calculated based on the resulting traffic states (typically traffic density, flow, and speed), and used to evaluate the performance of the traffic system under the impact of topology changes and traffic control measures (see Figure 4.1). From the traffic management point of view, the objectives of the co-design problem can vary from avoiding traffic congestion to increasing network safety and reliability, and to decreasing fuel consumption and pollution, etc. In this chapter, we consider a total monetary cost that includes the budget of construction and maintenance for the network, and a valuation of travel time and travel distance. Note that the construction cost is spent only once, but the maintenance cost is spent every year. Moreover, the price level of the maintenance is not constant but might change every year because of inflation effects. The value of travel time is often used for appraisal of road and public transport projects. It should be included in the monetary cost because it is closely related to the economical factors, e.g., drivers' wages, and interest or depreciation of the freight, etc. A full discussion of the value of travel time is out of the scope of this chapter, but we refer the interested reader to [41, 139] for more information on this topic. Moreover, travel distance is taken into account in this chapter as well because it is also related to the economic factors such as fuel consumption and wear of vehicles.

The main contributions of this chapter are:

1. We define a unified problem formulation for co-design of network topology and traffic control measures. We formulate the co-design problem in a model-based optimization framework, where the network topology design and traffic control measures are jointly applied to a traffic model, and a monetary cost is used to evaluate the performance of the traffic network;

2. We discuss four different solution frameworks for solving the proposed co-design problem, namely separate optimization, iterative optimization, bi-level optimization, and joint optimization, according to different requirements regarding performance and computational speed.

The rest of this chapter is structured as follows. Section 4.2 briefly summarizes the state-of-the-art on network topology design. Section 4.3 presents the problem statement. We formulate the co-design problem in a mathematical way in Section 4.4, and propose four

different solution frameworks in Section 4.5. A simulation-based case study of the Singapore freeway network is used to illustrate the proposed approach for solving the co-design problem in Section 4.6. Finally, we conclude our work in Section 4.7.

4.2 State-of-the-art on topology design

According to different forms of design decision variables, the network topology design problem can be posed in a discrete form that deals with changing the number of links or lanes, or a continuous form that deals changing the capacity of links in the network [145].

4.2.1 Discrete network topology design

The discrete network topology design problem concerns the modification of a traffic network by changing the numbers of links or lanes, and the design decision variables usually have binary or integer values. The objective of the discrete network topology design is to make an optimal investment decision in order to minimize both the total travel cost and total construction cost in the network. [96] presents a unified view for modeling the discrete network topology design problem, and proposes a unifying framework for describing a number of solution algorithms. [115] develops a bi-level programming method, where the lower level aims at finding a user equilibrium flow pattern in the traffic network, and the upper level then determines the design decision variables based on the equilibrium flow resulting from the lower level. [32] studies the lower-level problem by using a logit-based stochastic incremental traffic assignment approach. [61] proposes a new solution algorithm for the bi-level problem by using the support function concept. One of the challenges for the discrete network topology design problem is that one usually has to solve a nonlinear bi-level mixed-integer optimization problem, which could be extremely computationally complex.

4.2.2 Continuous network topology design

The continuous network topology design problem deals with improvement of the link capacity, and the design decision variables then have continuous values. From an implementation point view, there is not much difference between discrete network topology design and continuous network topology design. They have the same objective, and can be formulated in the same solution framework, e.g., bi-level programming. However, because the design decision variables are continuous in the continuous network topology design problem, the gradient of the objective function can be obtained, and gradient-based methods can be used to solve the proposed problem [33]. Moreover, sensitivity analysis methods [58, 135, 144] for the equilibrium network flows can then also be used.

4.3 Problem statement

Figure 4.2 shows the overall scheme of the co-design approach for determining the optimal network topology and the optimal traffic control strategies, involving two main parts — the real traffic system and the optimization layer.

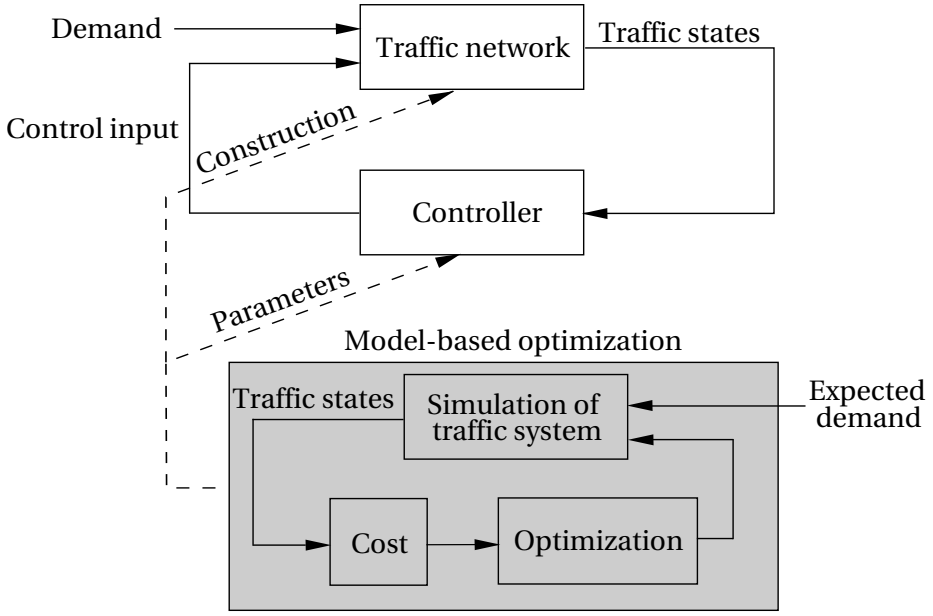


Figure 4.2: The overall co-design scheme.

The real traffic system contains the traffic network and the low-level controllers, such as traffic signals, ramp metering installations, variable speed limits, dynamic route information panels, and so on. The control measures are governed by parameterized control laws. The topology design decisions involve construction work in the network, e.g., building or removing lanes or links. Moreover, installing traffic controllers can also be considered as a topology design problem, e.g., whether to install the controllers, which type of the controllers to choose, and where to locate the controllers.

The optimization layer is used to determine the optimal network topology design decisions and the optimal parameters of the control laws. To do so, the real traffic system is simulated by traffic models in the optimization layer. The traffic simulation is based on:

- the expected daily demand patterns, as well as their long-term forecast;
- the planned topology design decisions and parameters of the control laws.

The performance is evaluated according to the simulated traffic states, and an optimization method is used to determine the optimal solutions. Usually, a traffic network is designed to be used for a long term. However, it will result in an extremely high computational burden to run the traffic simulation and optimization for the entire design period. Therefore, we choose a short range, e.g., one day or one month as a representative *basic design period*. The traffic simulation and optimization only run for the basic design period, but the resulting solutions are applied to the real traffic system for the entire design period, or for different time slots of the entire design period in the quasi-dynamic parameterized control approach (see Section 4.4.2).

4.4 Mathematical formulation

4.4.1 Network topology design

A vector $\boldsymbol{\delta}$ is defined as the construction decision variable. As mentioned in Section 4.2, the network topology design has a discrete form and a continuous form. For instance, the vector $\boldsymbol{\delta}$ in the discrete form can be defined as a decision variable corresponding to adding or removing links or lanes in the network. In this case, the mth element $\boldsymbol{\delta}_m$ of $\boldsymbol{\delta}$ corresponds to the number of lanes that will be added to or subtracted from link[1] m, where $\boldsymbol{\delta}_m$ with a positive value means addition of lanes, and $\boldsymbol{\delta}_m$ with a negative value means removal of lanes. In this chapter, we assume that the construction decision variable $\boldsymbol{\delta}_m$ may have a negative value, which means that links or lanes can be blocked in order to improve the performance of the network[2]. Note that the value of $\boldsymbol{\delta}_m$ is constrained by the current number of lanes on link m and the available free space: $\boldsymbol{\delta}_m^{\min} \leqslant \boldsymbol{\delta}_m \leqslant \boldsymbol{\delta}_m^{\max}$. More specifically, the lower bound value $\boldsymbol{\delta}_m^{\min}$ should guarantee that the total number of lanes on link m after construction is non-negative, and the upper bound value $\boldsymbol{\delta}_m^{\max}$ should guarantee that the total number of lanes on link m after construction does not exceed the physical limitations. Similarly, for the continuous form, the vector $\boldsymbol{\delta}$ corresponds to e.g. expansion or reduction of link capacity, in which case $\boldsymbol{\delta}_m$ could correspond to difference between the current width and the new width of lanes in link m.

Changing the network topology will in general influence the traffic flow patterns in the network, which means that the traffic state x (usually traffic flow, density, and speed) depends on the construction decision variable $\boldsymbol{\delta}$. Thus, traffic evolution can be described by a difference equation as follows:

$$x(k+1) = f(x(k), u(k), d(k), \boldsymbol{\delta}) \,, \tag{4.1}$$

where f is the traffic update function, $x(k) \in \mathbb{R}^{n_x}$ denotes the traffic state vector at simulation step k, $u(k) \in \mathbb{R}^{n_u}$ denotes the control inputs vector at simulation step k, and $d(k) \in \mathbb{R}^{n_d}$ denotes the disturbance (typically, the traffic demand) at simulation step k.

4.4.2 Parameterized traffic control

A parameterized traffic control law is generally formulated as:

$$u_{\mathrm{c}}(k_{\mathrm{c}}) = h(x(Mk_{\mathrm{c}}), d(Mk_{\mathrm{c}}), \boldsymbol{\theta}) \,, \tag{4.2}$$

where h denotes the control law, $u_{\mathrm{c}}(k_{\mathrm{c}}) \in \mathbb{R}^{n_u}$ denotes the control inputs vector at control step k_{c}, and $\boldsymbol{\theta} \in \mathbb{R}^{n_{\boldsymbol{\theta}}}$ contains all the parameters of the traffic controllers. The integer M is defined as by $M = T_{\mathrm{c}}/T$, with T_{c} the control interval and T the simulation interval. For the sake of simplicity, we assume that T is an integer divisor of T_{c}. We make an explicit difference between the control interval T_{c} and the simulation interval T, because traffic evolution is a comparatively fast process, while the control actions are not necessarily updated as fast as the traffic evolution. Therefore, the relationship between the control inputs $u_{\mathrm{c}}(k_{\mathrm{c}})$ and $u(k)$

[1] Link m can initially be a virtual link with 0 lanes. In that case, a new link m is considered to be built with $\boldsymbol{\delta}_m$ lanes ($\boldsymbol{\delta}_m \geqslant 0$).

[2] Braess et al. [23] illustrate that adding new links may in some cases deteriorate the performance of the traffic network.

is captured as $u(k) = u_c(\lfloor k/M \rfloor)$, where the operator $\lfloor\cdot\rfloor$ denotes the largest integer less than or equal to the function argument.

Two examples of the parameterized traffic control measures for the discrete-time, macroscopic model are formulated in this section, and will be used for the case study in Section 4.6. Note that one can design a parameterized control law for any type of control measure, and that the formulation of the control law does not necessarily have to follow the structures to be presented here.

- A possible control law for ramp metering originates from ALINEA[3] [108]:

$$r_o(k_c) = \min\left(\max\left(r_o(k_c-1) + \boldsymbol{\theta}_o^r \frac{\rho_{\text{crit},m} - \rho_{m,1}(Mk_c)}{\rho_{\text{crit},m}}, 0\right), 1\right), \tag{4.3}$$

  where $r_o(k_c)$ denotes the ramp metering rate at the origin o connected to link m at control step k_c, $\rho_{m,1}(Mk_c)$ denotes the density of the first segment of link m at the simulation step $k = Mk_c$, $\rho_{\text{crit},m}$ denotes the critical density on link m, and $\boldsymbol{\theta}_o^r$ is the metering parameter to be designed. A high value of $\boldsymbol{\theta}_o^r$ means that the ramp metering rate is more sensitive to the change in the traffic density. The functions $\max(\cdot)$ and $\min(\cdot)$ are used to bound the ramp metering rate between 0 and 1.

- A possible control law for variable speed limits is formulated as follows [147]:

$$\begin{aligned} v_{m,i}^{\lim}(k_c) = \min\Big(\max\Big(&\boldsymbol{\theta}_{m,0}^v v_{m,i}^{\lim}(k_c-1) + \boldsymbol{\theta}_{m,1}^v \frac{v_{m,i+1}(Mk_c) - v_{m,i}(Mk_c)}{v_{m,i+1}(Mk_c) + \kappa_{m,v}} \\ &+ \boldsymbol{\theta}_{m,2}^v \frac{\rho_{m,i+1}(Mk_c) - \rho_{m,i}(Mk_c)}{\rho_{m,i+1}(Mk_c) + \kappa_{m,\rho}}, v_m^{\min}\Big), v_{\text{free},m}\Big), \end{aligned} \tag{4.4}$$

  where $v_{m,i}^{\lim}(k_c)$ denotes the maximum allowed speed of segment i on link m at control step k_c, and $\kappa_{m,v}$ and $\kappa_{m,\rho}$ are parameters preventing denominators from becoming zero. The constants $\boldsymbol{\theta}_{m,0}^v$, $\boldsymbol{\theta}_{m,1}^v$, and $\boldsymbol{\theta}_{m,2}^v$ are the target parameters to be designed. The maximum allowed variable speed $v_{m,i}^{\lim}(k_c)$ is bounded by the free flow speed $v_{\text{free},m}$ and the minimum speed $v_m^{\min}$.

In principle, the parameter vector $\boldsymbol{\theta}$ should be constant for a relatively long time, i.e., at the same time scale as the topology design. However, simple static traffic control using the same parameters over a long period may be inadequate for dynamic demand. In order to improve the performance of the traffic controllers, we consider a so-called quasi-dynamic parameterized traffic control approach. As shown in Figure 4.3, the entire design period is divided into years, a year can be divided into seasons, a season can be divided into days, and a day can be even further divided into different time slots, e.g., morning rush hours, midday non-rush hours, afternoon rush hours, and evening non-rush hours. We assume that driver behavior and traffic situations such as weather condition change in different seasons, so a

[3] In ALINEA, the occupancy, which is defined as the fraction of time that vehicles are detected by the sensors, is used to determine the ramp metering rate, while in our approach, the traffic density is used in the control law instead. The reason is that the traffic density is usually used as a state variable in many macroscopic traffic models; however, it is not easy to directly measure the density via the sensors used in traffic networks (in particular loop detectors). Hence, the occupancy is often used in practice. Note however that there is an approximate relationship between the traffic density and the occupancy [4]: occupancy = density × average vehicle length.

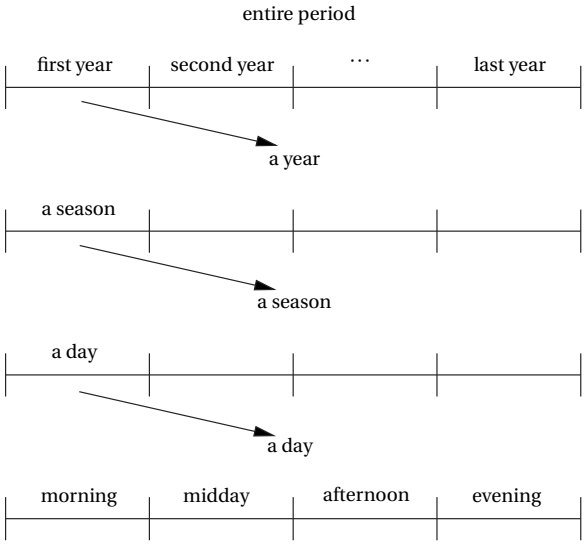


Figure 4.3: Sub-periods for quasi-dynamic parameterized traffic control.

season is considered as the basic unit for a set of parameters in the quasi-dynamic parameterized control. For one day, the parameters in different time slots have different values, while for the same time slot, the parameters in different days have the same value. In this way, we only need to optimize the parameters for one day, but can apply them for a season. In such a way, for complicated traffic situations, the parameters can be defined in a much more refined way than in the case of static traffic control.

4.4.3 Traffic models

The co-design problem uses a model-based optimization approach to determine the topology design decisions and the parameters of the traffic control laws. The traffic model used in this approach consists of two parts: a traffic flow model, and a route choice model. In this chapter, as an example, we choose the METANET model [98] as the traffic flow model, and the multinomial logit model [121] as the route choice model. However, it is important to note that our approach is generic, so any other traffic flow model and route choice model can replace the ones used here.

As we have a network with multiple destinations involving route choice, we select the destination-dependent version of METANET model. The destination-dependent METANET model has the following characteristics:

- The traffic network is divided into links that corresponds to homogeneous freeway stretches, and each link m is divided into N_m segments of equal length L_m (typically 500-1000 m);

- The traffic network has three different types of nodes, origins, destinations, and intermediate nodes, and let $\mathcal{O}$, $\mathcal{D}$ and $\mathcal{N}$ be the set of origins, destinations and intermediate nodes respectively;

- Each segment i of link m is characterized by its density $\rho_{m,i,d}(k)$ (veh/km/lane), mean speed $v_{m,i}(k)$ (km/h), outflow $q_{m,i}(k)$ (veh/h) at simulation step k, and number of lanes λ_m;
- Each origin is described by a traffic waiting queue with length $w_{o,d}(k)$ (veh);
- At each segment, the composition rate $\gamma_{m,i,d}(k)$ denotes the fraction of traffic flow to destination d on segment i of link m at simulation step k;
- At each intermediate node, the splitting rate $\beta_{n,m,d}(k)$ expresses the fraction of the total flow with destination d that leaves node n via outgoing link m at simulation step k.

The main equations of the METANET model can be found in Appendix B.

The route choice behavior of drivers in the METANET model is described by the splitting rate $\beta_{n,m,d}(k)$. In our case, the splitting rate $\beta_{n,m,d}(k)$ is determined by the travel time according to the logit model:

$$\beta_{n,m,d}(k) = \frac{e^{-\xi_{n,d} t_{n,m,d}(k)}}{\sum_{m' \in \mathscr{O}_n} e^{-\xi_{n,d} t_{n,m',d}(k)}}, \tag{4.5}$$

where $t_{n,m,d}(k)$ denotes the predicted travel time from node n to destination d via link m at simulation step k, and $\mathscr{O}_n$ denotes the set of outgoing links from node n. The predicted travel time $t_{n,m,d}(k)$ can be communicated to the drivers via variable message signs, on-board devices, or on-line traffic information (e.g., radio or other resources). The parameter $\xi_{n,d} > 0$ reflects how drivers react on a travel time difference between different routes to destination d at node n. The higher $\xi_{n,d}$, the less travel time difference is needed to convince drivers to choose the fastest route.

4.4.4 Performance criteria

The objective of the co-design problem is two-fold:

- To improve the traffic flows in the network;
- To reduce the construction and maintenance cost.

Flow performance criteria

As an illustration, let us interpret improving the traffic flows as minimizing both the total time spent (TTS) and total distance traveled (TDT) by all the vehicles in the network. We define a day as the representative basic design period. The TTS and the TDT for the given day is formulated by a monetary valuation, and can be easily calculated for METANET as follows:

$$J_{\mathrm{TTS}} = \sum_{k \in \mathscr{K}} \sum_{d \in \mathscr{D}} \Big(\sum_{m \in \mathscr{M}} \sum_{i \in \mathscr{I}_m} \alpha^{\mathrm{t}} \rho_{m,i,d}(k) L_m \lambda_m T + \sum_{o \in \mathscr{H}} \alpha^{\mathrm{w}} w_{o,d}(k) T \Big) + J_{\mathrm{TTS,endpoint}}, \tag{4.6}$$

$$J_{\mathrm{TDT}} = \sum_{k \in \mathscr{K}} \sum_{m \in \mathscr{M}} \sum_{i \in \mathscr{I}_m} \alpha^{\mathrm{d}} q_{m,i}(k) T L_m + J_{\mathrm{TDT,endpoint}}, \tag{4.7}$$

with α^{t} (\$/h) the monetary cost per unit travel time, α^{w} (\$/h) the monetary cost per unit waiting time, α^{d} (\$/km) the monetary cost per unit distance, $w_{o,d}(k)$ the number of vehicles to destination d waiting on origin o at simulation step k, $\mathscr{M}$ the set of links of the network, $\mathscr{I}_m$ the set of segments on link m, $\mathscr{H}$ the set of origins, $\mathscr{K}$ the set of simulation steps for the given day, and $J_{\mathrm{TTS,endpoint}}$ and $J_{\mathrm{TDT,endpoint}}$ are end point penalties. The factor $\rho_{m,i,d}(k)L_m\lambda_m$ in (4.6) indicates the number of vehicles with destination d in segment i of link m, and hence multiplied by the time interval T this gives the time spent by the vehicles in the corresponding segment. Similarly, the term $q_{m,i}(k)T$ in (4.7) represents the number of vehicles leaving segment i of link m, and multiplied by the length of segment L_m this gives the distance traveled by the vehicles. Note that because we consider a representative day as the basic design period, it is possible that some vehicles may be still traveling in the network, or even waiting at origins at the end of the day. These vehicles will eventually reach their destination, and hence we should also take the travel time cost and travel distance cost spent after the end of the representative day into account, and add them to J_{TTS} and J_{TDT} respectively. This part of the cost is called an end point penalty [95]. The end point penalties $J_{\mathrm{TTS,endpoint}}$ and $J_{\mathrm{TDT,endpoint}}$ can be computed in several ways, e.g.,

1. We estimate the TTS and the TDT spent by all the vehicles that are still in the network or at the origins after the end of the representative day:

$$J_{\mathrm{TTS,endpoint}} = \alpha^{\mathrm{t}} \sum_{d\in\mathscr{D}} \Big(\sum_{m\in\mathscr{M}} \sum_{i\in\mathscr{I}_m} \rho_{m,i,d}(k_{\mathrm{end}}) L_m \lambda_m \tau_{m,i,d} + \sum_{o\in\mathscr{H}} w_{o,d}(k_{\mathrm{end}}) \tau_{o,d} \Big), \tag{4.8}$$

$$J_{\mathrm{TDT,endpoint}} = \alpha^{\mathrm{d}} \sum_{d\in\mathscr{D}} \Big(\sum_{m\in\mathscr{M}} \sum_{i\in\mathscr{I}_m} \gamma_{m,i,d}(k_{\mathrm{end}}) q_{m,i}(k_{\mathrm{end}}) T \ell_{m,i,d} + \sum_{o\in\mathscr{H}} w_{o,d}(k_{\mathrm{end}}) \ell_{o,d} \Big), \tag{4.9}$$

   with $\tau_{m,i,d}$ and $\tau_{o,d}$ the typical travel times that a vehicle in the segment i of link m and at origin o needs to reach destination d, $\ell_{m,i,d}$ and $\ell_{o,d}$ the typical travel distances that a vehicle in the segment i of link m and at origin o needs to reach destination d, and $k_{\mathrm{end}} \in K$ the last simulation step of K. The typical travel time and distance can be calculated based on historical data or a prediction model.

2. We use the same expressions of (4.8) and (4.9) to calculate the end point penalties, but the typical travel time $\tau_{m,i,d}(k_{\mathrm{end}})$ and $\tau_{o,d}(k_{\mathrm{end}})$ and the typical travel distance $\ell_{m,i,d}$ and $\ell_{o,d}$ are determined according to the traffic state on the fastest or shortest route at simulation step k_{end}:

$$\tau_{m,i,d}(k_{\mathrm{end}}) = \sum_{j=i+1}^{N_m} \frac{L_m}{v_{m,j}(k_{\mathrm{end}})} + \sum_{l\in r_{m,d}} \sum_{j=1}^{N_l} \frac{L_l}{v_{l,j}(k_{\mathrm{end}})}, \tag{4.10}$$

$$\tau_{o,d}(k_{\mathrm{end}}) = \sum_{l\in r_{o,d}} \sum_{j=1}^{N_l} \frac{L_l}{v_{l,j}(k_{\mathrm{end}})}, \tag{4.11}$$

$$\ell_{m,i,d} = \sum_{j=i+1}^{N_m} L_m + \sum_{l\in r_{m,d}} \sum_{j=1}^{N_l} L_l, \tag{4.12}$$

$$\ell_{o,d} = \sum_{l \in r_{o,d}} \sum_{j=1}^{N_l} L_l \,, \tag{4.13}$$

where $r_{m,d}$ denotes the fastest or shortest route from the end of link m to destination d, and $r_{o,d}$ denotes the fastest or shortest route from origin o to destination d.

3. We determine K shortest or fastest loop-less routes [68, 79, 146] from any link end and any origin to any destination — provided that some vehicles are still traveling between that link or origin and that destination at the end of the representative day — and then those vehicles are distributed over these K shortest routes. The end point penalties are then computed by summing the total travel time cost and the total travel distance cost for all the vehicles remaining in the network at simulation step k_{end}.

4. We keep simulating the traffic network until all the vehicles have left it, while setting the demand to zero after the end of the representative day. The end point penalties can then be directly computed based on the simulation results.

One should choose one of these four approaches w.r.t finding a balanced trade-off between the accuracy and the computation speed when computing the end point penalties.

For each sub-period (e.g., week, season, year) corresponding to the quasi-dynamic approach of Section 4.4.2 and Figure 4.3, by computing the TTS and the TDT for those days using (4.6) and (4.7), and next multiplying the result with the number of days of the given sub-period, we can compute yearly monetary TTS and TDT values $J_{\text{TTS},y}$ and $J_{\text{TDT},y}$ for each year y in the full period under consideration.

Construction and maintenance performance criteria

Another objective is to reduce the construction and maintenance cost of the network. A linear construction cost function can be adopted:

$$J_{\text{CC}} = \sum_{\substack{m \in \mathcal{M} \\ \boldsymbol{\delta}_m > 0}} \alpha^{\text{c}}_m N_m L_m \boldsymbol{\delta}_m - \sum_{\substack{m \in \mathcal{M} \\ \boldsymbol{\delta}_m < 0}} \alpha^{\text{r}}_m N_m L_m \boldsymbol{\delta}_m \,, \tag{4.14}$$

where α^{c}_m (\$/year) denotes the construction cost per lane per unit length, α^{r}_m (\$/year) denotes the removal cost per lane per unit length, and N_m denotes the number of segments on link m. The maintenance cost depends on the number of lanes after construction, and the cost for the first year is formulated as:

$$J_{\text{MC}} = \sum_{m \in \mathcal{M}} \alpha^{\text{m}}_m N_m L_m (\lambda_m + \boldsymbol{\delta}_m) \,, \tag{4.15}$$

where α^{m}_m (\$/year) denotes the maintenance cost per lane per unit length.

Overall performance criteria

The overall objective function can be obtained by summing up all the objective functions. Note that the major difference between the construction cost and the other three costs is that the construction cost is only spent in the first year, while the other three costs should be considered every year, and will be increasing annually due to the inflation effect. We assume

that the inflation effect starts at the beginning of every year, so the overall objective function is formulated as:

$$J = J_{\mathrm{CC}} + \sum_{y=1}^{N_{\mathrm{y}}} (1+r)^{y-1} \left(J_{\mathrm{MC}} + J_{\mathrm{TTS},y} + J_{\mathrm{TDT},y} \right), \tag{4.16}$$

where r denotes the yearly inflation rate, and N_{y} denotes the number of years during the entire design period.

4.4.5 Constraints

Recall that as introduced in Section 4.4.1 the value of the topology design decision variable $\boldsymbol{\delta}_m$ is constrained by the number of lanes on link m and the available free space: $\boldsymbol{\delta}_m^{\min} \leqslant \boldsymbol{\delta}_m \leqslant \boldsymbol{\delta}_m^{\max}$. Moreover, in order to avoid reducing the region of possible solutions, or causing an infeasible problem, constraints on the parameters of control laws are not directly put on the parameters themselves, but instead a minimum and a maximum value is added to each control signal as shown in (4.3) and (4.4). Moreover, the traffic simulation model is also included as a set of equality constraints.

4.4.6 Overall optimization problem

So far we have introduced the objective functions and the constraints for optimization, together with the topology design methods, traffic control measures, and traffic models. The overall optimization problem can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{\theta},\boldsymbol{\delta}} \quad & J(\boldsymbol{\theta},\boldsymbol{\delta}) \\ \text{subject to} \quad & y(\boldsymbol{\theta},\boldsymbol{\delta}) = 0 \\ & z(\boldsymbol{\theta},\boldsymbol{\delta}) \leqslant 0 \end{aligned} \tag{4.17}$$

where $J(\cdot)$ includes all the objective functions, $y(\cdot) = 0$ includes all the equality constraints, and $z(\cdot) \leqslant 0$ includes all the inequality constraints. For the sake of compactness in Section 4.5, the constraints of (4.17) will be denoted via a constraint set $\mathscr{C}$

$$\mathscr{C} = \{(\boldsymbol{\theta},\boldsymbol{\delta}) | y(\boldsymbol{\theta},\boldsymbol{\delta}) = 0 \text{ and } z(\boldsymbol{\theta},\boldsymbol{\delta}) \leqslant 0\}. \tag{4.18}$$

In addition, we define $\mathscr{C}_{\boldsymbol{\theta}}(\boldsymbol{\delta})$ and $\mathscr{C}_{\boldsymbol{\delta}}(\boldsymbol{\theta})$ as the sets of feasible $\boldsymbol{\theta}$ and $\boldsymbol{\delta}$ for a given value of $\boldsymbol{\delta}$ and $\boldsymbol{\theta}$ respectively, i.e.,

$$\mathscr{C}_{\boldsymbol{\theta}}(\boldsymbol{\delta}) = \{\boldsymbol{\theta} | (\boldsymbol{\theta},\boldsymbol{\delta}) \in \mathscr{C}\} \tag{4.19}$$

$$\mathscr{C}_{\boldsymbol{\delta}}(\boldsymbol{\theta}) = \{\boldsymbol{\delta} | (\boldsymbol{\theta},\boldsymbol{\delta}) \in \mathscr{C}\}. \tag{4.20}$$

4.5 Solution approaches

4.5.1 Solution frameworks

We have described an optimization formulation (4.17) in order to solve the co-design problem of the network topology and traffic control measures. However, optimizing both $\boldsymbol{\theta}$ and $\boldsymbol{\delta}$

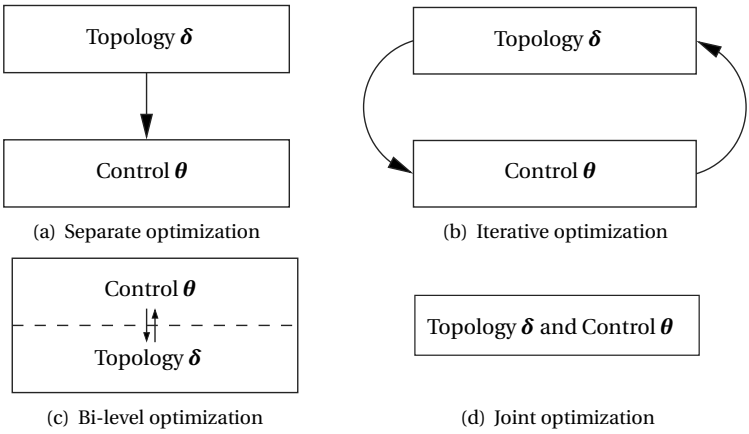


Figure 4.4: Schematic representation of four solution frameworks.

will potentially result in a high computational burden. Therefore, the development of an efficient solution approach is required to guarantee successful solution of the co-design problem.

We discuss four solution frameworks for (4.17), i.e., separate optimization, iterative optimization, bi-level optimization, and joint optimization (see Figure 4.4). We introduce each framework in more detail next.

Separate optimization

Separate optimization means that the optimization problem is decomposed into multiple separate subproblems. This framework has been applied to solve e.g. the machining optimization problem [122], and the mesh optimization problem [74]. In our case, we first determine the optimal topology design decision variable $\boldsymbol{\delta}$ by fixing the parameters of the traffic control law to $\boldsymbol{\theta}_{\text{fix}}$:

$$\boldsymbol{\delta}^* = \arg \min_{\boldsymbol{\delta} \in \mathscr{C}_{\boldsymbol{\delta}}(\boldsymbol{\theta}_{\text{fix}})} J(\boldsymbol{\theta}_{\text{fix}}, \boldsymbol{\delta}) \,. \tag{4.21}$$

The value of $\boldsymbol{\theta}_{\text{fix}}$ can be chosen according to the default settings of the control laws, or based on knowledge from experts. After $\boldsymbol{\delta}^*$ is determined, we then only optimize the parameters $\boldsymbol{\theta}$ of the control laws in the network with the fixed topology:

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta} \in \mathscr{C}_{\boldsymbol{\theta}}(\boldsymbol{\delta}^*)} J(\boldsymbol{\theta}, \boldsymbol{\delta}^*) \tag{4.22}$$

This yields the pair $(\boldsymbol{\delta}^*, \boldsymbol{\theta}^*)$ as the approximate solution of the optimization problem (4.17).

Iterative optimization

A more advanced approach compared to separate optimization is iterative optimization. In iterative optimization, we also solve the optimization problems (4.21) and (4.22), but these two problems are solved iteratively instead of only once. The solution $\boldsymbol{\theta}^*$ from (4.22) is fed

back to (4.21) by replacing $\boldsymbol{\theta}_{\text{fix}}$ with $\boldsymbol{\theta}^*$, and in this way, a new solution of $\boldsymbol{\delta}^*$ can be obtained. Generally, we repeat solving:

$$\boldsymbol{\delta}^{*^{i+1}} = \arg \min_{\boldsymbol{\delta} \in \mathscr{C}_{\boldsymbol{\delta}}\left(\boldsymbol{\theta}^{*^i}\right)} J\left(\boldsymbol{\theta}^{*^i}, \boldsymbol{\delta}\right) \tag{4.23}$$

$$\boldsymbol{\theta}^{*^{i+1}} = \arg \min_{\boldsymbol{\theta} \in \mathscr{C}_{\boldsymbol{\theta}}\left(\boldsymbol{\delta}^{*^{i+1}}\right)} J\left(\boldsymbol{\theta}, \boldsymbol{\delta}^{*^{i+1}}\right) \tag{4.24}$$

until one of the following stop criteria is satisfied:

1. The maximum number of iteration steps N is reached;
2. The difference in the values of $\boldsymbol{\delta}$ and $\boldsymbol{\theta}$ between two consecutive iteration steps is smaller than some predefined threshold: $\|\boldsymbol{\delta}^{*^{i+1}} - \boldsymbol{\delta}^{*^i}\| < \epsilon_{\boldsymbol{\delta}}$ and $\|\boldsymbol{\theta}^{*^{i+1}} - \boldsymbol{\theta}^{*^i}\| < \epsilon_{\boldsymbol{\theta}}$.

More information about the iterative optimization approach can be found in [13, 22].

Bi-level optimization

Bi-level optimization [28] divides the optimization problem (4.17) into two levels (an inner one and an outer one), where one problem is embedded within another. Generally, the outer optimization task is commonly referred to as the upper-level optimization task, and the inner optimization task is commonly referred to as the lower-level optimization task. In the given problem setting, it is assumed that for any network topology (a given $\boldsymbol{\delta}$), there exists a corresponding optimal parameter $\boldsymbol{\theta}^*(\boldsymbol{\delta})$ of the control law. Therefore, in the bi-level optimization, the lower level aims at finding a relationship between $\boldsymbol{\delta}$ and $\boldsymbol{\theta}$, and the upper level aims at determining the network topology.

The relationship between $\boldsymbol{\delta}$ and $\boldsymbol{\theta}$ can be expressed by a function $\boldsymbol{\theta}^*(\boldsymbol{\delta})$, which can be obtained from the lower level optimization

$$\boldsymbol{\theta}^*(\boldsymbol{\delta}) = \arg \min_{\boldsymbol{\theta} \in \mathscr{C}_{\boldsymbol{\theta}}(\boldsymbol{\delta})} J(\boldsymbol{\theta}, \boldsymbol{\delta}) \,. \tag{4.25}$$

In the upper level, the topology design decision is determined according to

$$\boldsymbol{\delta}^* = \arg \min_{\boldsymbol{\delta} \in \mathscr{C}_{\boldsymbol{\delta}}} J(\boldsymbol{\theta}^*(\boldsymbol{\delta}), \boldsymbol{\delta}) \,, \tag{4.26}$$

with $\mathscr{C}_{\boldsymbol{\delta}} = \{\boldsymbol{\delta} | \exists \boldsymbol{\theta}$ such that $(\boldsymbol{\theta}, \boldsymbol{\delta}) \in \mathscr{C}\}$. The optimal parameter is obtained by:

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}^*(\boldsymbol{\delta}^*) \tag{4.27}$$

Joint optimization

The aforementioned three frameworks deal with the topology design and traffic control measures one after the other. Unlike them, joint optimization solves (4.17) by considering both $\boldsymbol{\delta}$ and $\boldsymbol{\theta}$ at the same time, which can be obtained by:

$$(\boldsymbol{\theta}^*, \boldsymbol{\delta}^*) = \arg \min_{(\boldsymbol{\theta}, \boldsymbol{\delta}) \in \mathscr{C}} J(\boldsymbol{\theta}, \boldsymbol{\delta}) \tag{4.28}$$

Discussion

One should consider the complexity of the co-design problem when choosing the appropriate solution framework. The complexity of the problem changes when the design settings are different. For instance, choosing a week as the basic design period, in which the expected traffic demand and traffic situations can be distinguished between weekdays and weekends, can yield more refined design results than choosing a day as the basic design period. However, the computational burden for simulation will be also higher when a longer basic design period. Among the aforementioned solution frameworks, the separate optimization approach is the easiest approach to implement. The iterative optimization approach will in general achieve a better performance than the separate optimization approach, but the computation speed may be lower because of the iterative procedure. [29] presents the iterative optimization approach to solve a co-design problem with network topology and traffic signal setting. To the best knowledge of the authors, the co-design problem by using the bi-level optimization approach or the joint optimization approach has not yet been investigated in the literature. In these two approaches, the topology design and the traffic control can interact with each other well, so the performance is expected to be improved significantly. However, the computation time will also increase highly. Therefore, an appropriate solution framework should be selected to provide a balance between the performance and the computation speed according to the design requirements.

4.5.2 Optimization algorithms

In general, the co-design problem will result in a non-linear, non-convex optimization problem. To tackle such a problem, different optimization algorithms can be applied.

- For real-valued problems, i.e., when the topology design has a continuous form and the parameters of the control law have continuous values, we can use multi-start sequential quadratic programming [19], pattern search [73], simulated annealing [83], genetic algorithms [5], and so on;
- For mixed-integer problems, i.e., when the topology design has a discrete form or the parameters of the control law have discrete values, we can also use genetic algorithms and simulated annealing. Moreover, other mixed-integer nonlinear programming approaches such as branch-and-bound algorithms [89], branch-and-cut algorithms [133], Benders decomposition method [63], and outer approximation method [48] can be used as well.

4.6 Case study

4.6.1 Set-up

We illustrate the four proposed solution frameworks to the co-design problem introduced in Section 4.5.1 in a simulation of the central and eastern parts of the Singapore expressway network (Figure 4.5), which is one of the busiest areas in Singapore, including the central business district and the international airport. This area contains 40 one-way highway links (with the parameters of each link presented in Table 4.1), 8 origins (o_1–o_8), and 8 destinations (d_1–d_8).

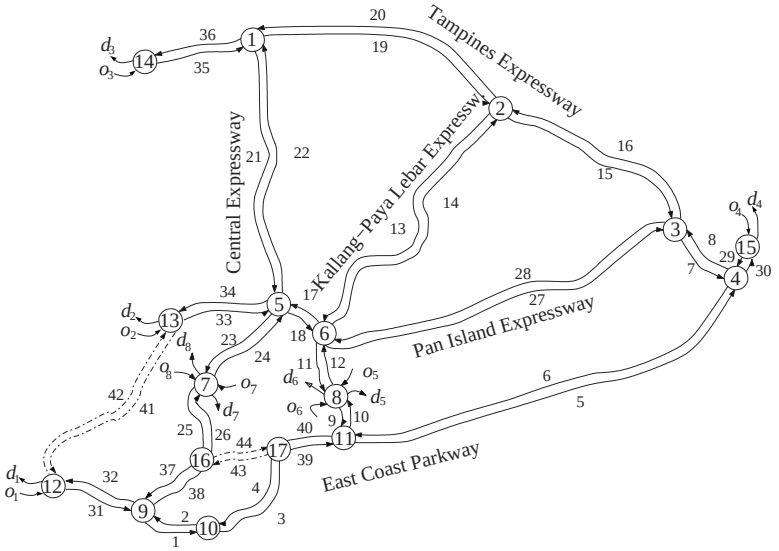


Figure 4.5: The central and eastern parts of the Singapore expressway network.

Table 4.1: Parameters of links in the Singapore expressway network

Link index	Length (km)	Number of lanes	Nodes	Capacity (veh/h)
1, 2	3.0	3	9, 10	6000
3, 4	3.5	4	10, 17	8000
5, 6	13.0	4	11, 4	8000
7, 8	2.0	4	3, 4	8000
9, 10	1.0	3	8, 11	6000
11, 12	2.0	3	6, 8	6000
13, 14	8.0	3	2, 6	6000
15, 16	6.5	3	2, 3	6000
17, 18	2.0	4	6, 5	8000
19, 20	7.0	3	1, 2	6000
21, 22	7.5	2	1, 5	4000
23, 24	3.5	2	5, 7	4000
25, 26	2.5	2	7, 16	4000
27, 28	11.0	4	6, 3	8000
29, 30	1.0	4	15, 4	8000
31, 32	3.0	3	12, 9	6000
33, 34	3.0	4	13, 5	8000
35, 36	3.0	3	14, 1	6000
37, 38	2.0	2	9, 16	4000
39, 40	1.0	4	11, 17	8000

The topology design problem involves determining whether new links should be constructed or existing links should be removed, or whether the number of lanes on the existing links should be changed. According to the simulation scenario (see Section 4.6.3), we assume that new links 41 and 42 can be built between nodes 12 and 13, and new links 43 and 44 can be built between nodes 16 and 17, as shown by dash lines in Figure 4.5. The upper bound on the number of lanes for each new link is set as 2. Moreover, we may change the number of lanes on links 9, 17, 23, 25, 32, and 33, which are main roads around the central business district. The upper bound for the number of lanes for each existing link is set as 4. As traffic control measures, ramp metering installations are put at origins o_2, o_4, o_5, and o_7, and variable speed limits are installed on link 32, as well as link 41 if it is built. Only the segments on the second half of link 32 and 41 are controlled by the dynamic speed limits, which is similar in the settings considered by [72]. Eventually, we have a co-optimization problem with 10 integer optimization variables and 10 continuous optimization variables.

4.6.2 Optimization and model parameters

According to a report by the [137], the unit cost of building a stretch of highway ranged from about \$2.5 million per km to \$16 million per km in 25 U.S. states in 2002. In this case study, we set the construction cost as $\alpha_m^{\mathrm{c}} = \$10$ million per lane per km, and the removal cost as $\alpha_m^{\mathrm{r}} = \$5$ million per lane per km. The maintenance cost for the first year is $\alpha_m^{\mathrm{m}} = \$1$ million per lane per km, and it will increase every year at the rate of $r = 0.04$. The total length of the entire design period is set to $N_{\mathrm{y}} = 20$ years. For other parameters, we assume that the travel time cost is equal to the waiting time cost, defined as $\alpha^{\mathrm{t}} = \alpha^{\mathrm{w}} = \$10$ per h per vehicle, and the travel distance cost is set as $\alpha^{\mathrm{d}} = \$1$ per km per vehicle.

The destination-dependent METANET model is used to simulate the traffic flow evolution, and we use the traffic control laws (4.3) and (4.4) in this case study. The model parameters are defined as follows [71]: simulation time step length $T = 10$ s, free flow speed $v_{\mathrm{free},m} = 120$ km/h, lower bound of speed limit $v_m^{\min} = 50$ km/h, critical density $\rho_{\mathrm{crit},m} = 33.5$ veh/km/lane, maximum density $\rho_{\mathrm{max},m} = 180$ veh/km/lane, flow capacity $q_{\mathrm{cap}} = 2000$ veh/h/lane. Other parameters can be found in [86].

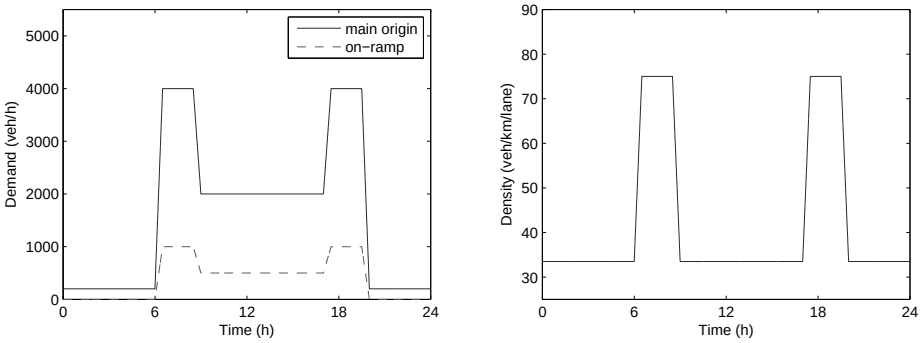
4.6.3 Scenario

We consider four different traffic inflows in the network (see Figure 4.6(a)) from origins o_2, o_4, o_5, and o_7, and all of them have the same destination d_1:

- For traffic flows from o_2 and o_4, we define a high traffic demand profile: it starts with a flow of 200 veh/h at 12.00 a.m., reaches the first peak of 4000 veh/h at 6.30 a.m., gradually decreases to 2000 veh/h at 10.00 a.m., reaches the second peak of 4000 veh/h at 5.30 p.m., decreases to 200 veh/h at 9.00 p.m., and maintains this level until midnight;

- For traffic flows from o_5 and o_7, we define a low traffic demand profile: it starts with a flow of 0 veh/h at 12.00 a.m., reaches the first peak of 1000 veh/h at 6.30 a.m., decreases to 100 veh/h at 10.00 a.m., reaches the second peak of 1000 veh/h at 5.30 p.m., decreases to 0 veh/h at 9.00 p.m., and maintains this level until midnight.

In order to create traffic congestion during the peak hours, we add a large pulse with a maximum value of 75 veh/km/lane to the downstream density of the network at each peak-hour



(a) Demand profile (b) Downstream density profile

Figure 4.6: Traffic scenario considered in the simulation experiment.

period, as shown in Figure 4.6(b). These pulses can generate a back-propagating wave and make the traffic in the network more busy.

4.6.4 Simulation results

We use the four solution frameworks to solve the co-design problem for the Singapore expressway network, and compare the results. In order to prevent that the results from different solution frameworks would be fully determined by the optimization algorithm, two optimization algorithms are used to benchmark all the solution frameworks: one is a genetic algorithm implemented in the `ga` function of the Matlab Global Optimization Toolbox, and the other is an outer approximation branch-and-bound algorithm implemented in the `minlp` function of the TOMLAB Mixed-Integer Nonlinear Programming toolbox for Matlab. The default parameters settings of the `ga` function are used in this case study. More information about parameter tunning of genetic algorithms can be found in [50, 94].

The results of the four solution frameworks by using `ga` and `minlp` are displayed in Tables 4.2 and 4.3. We can see that for each solution framework the total cost is almost the same by using the `ga` and `minlp` functions, except that in the separate optimization framework, the total cost by using the `minlp` function is a little higher than the one by using the `ga` function. Therefore, we can conclude that the simulation results are mainly determined by the solution framework, not by the optimization algorithms. Moreover, it clearly shows that the separate optimization framework results in a much higher total cost than the other three frameworks. The other three frameworks yield almost the same solution in this case study, especially for the total cost and for the topology design decisions. In Tables 4.2 and 4.3, for the construction decision $\boldsymbol{\delta}$, the subscript of each variable corresponds to the index of the link, and the value of each variable indicates the modification of the numbers of lanes on the link, i.e., positive values mean adding lanes, negative values mean removing lanes, and zero values mean no change. For the control parameter $\boldsymbol{\theta}$, the superscript 'r' means that the variable is a parameter for the on-ramp control law, and the superscript 'v' means that the variable is a parameter for the variable speed limits control law. Since the variable speed limits control law has three parameters, we use 0, 1, and 2 in the subscript to distinguish these three parameters. In Table 4.2, the joint optimization yields different topology design

Table 4.2: Simulation results obtained by using `ga`

Solution framework	Total cost ($)	Topology decision	Control parameters
Separate optimization	$4.6431 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -2$ $\boldsymbol{\delta}_{12} = -2$ $\boldsymbol{\delta}_{17} = -4$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 2$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 2$ $\boldsymbol{\delta}_{44} = 0$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 0.61$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 0.73$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 0.10$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 1.65$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.37$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 2.36$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 0.04$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.92$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 1.78$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 2.80$
Iterative optimization	$3.6430 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -1$ $\boldsymbol{\delta}_{12} = -3$ $\boldsymbol{\delta}_{17} = -3$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 2$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 2$ $\boldsymbol{\delta}_{44} = 0$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 13.72$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 12.75$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 18.62$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 8.99$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.49$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 337.81$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 1127.17$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.62$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 104.37$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 2311.53$
Bi-level optimization	$3.6428 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -1$ $\boldsymbol{\delta}_{12} = -3$ $\boldsymbol{\delta}_{17} = -3$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 2$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 2$ $\boldsymbol{\delta}_{44} = 0$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 13.23$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 11.95$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 8.17$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 14.50$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.55$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 917.19$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 760.89$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.76$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 226.12$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 2858.39$
Joint optimization	$3.6399 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -3$ $\boldsymbol{\delta}_{12} = -2$ $\boldsymbol{\delta}_{17} = -4$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 0$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 0$ $\boldsymbol{\delta}_{44} = 1$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 17.20$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 13.21$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 10.94$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 10.53$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.19$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 315.06$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 264.98$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 0.73$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 375.66$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 2677.94$

Table 4.3: Simulation results obtained by using `minlp`

Solution framework	Total cost ($)	Topology decision	Control parameters
Separate optimization	$4.7390 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -2$ $\boldsymbol{\delta}_{12} = -2$ $\boldsymbol{\delta}_{17} = -4$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 2$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 2$ $\boldsymbol{\delta}_{44} = 0$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 1.97$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 0.73$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 0.01$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 0.35$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.47$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 0.06$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 0.91$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.12$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 1.77$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 0.86$
Iterative optimization	$3.6435 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -1$ $\boldsymbol{\delta}_{12} = -3$ $\boldsymbol{\delta}_{17} = -3$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 2$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 2$ $\boldsymbol{\delta}_{44} = 0$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 12.09$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 13.95$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 10.62$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 14.03$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.71$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 144.35$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 617.42$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.49$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 293.56$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 1848.33$
Bi-level optimization	$3.6435 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -1$ $\boldsymbol{\delta}_{12} = -3$ $\boldsymbol{\delta}_{17} = -3$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 2$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 2$ $\boldsymbol{\delta}_{44} = 0$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 13.08$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 8.86$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 10.62$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 16.01$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.03$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 285.77$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 1230.33$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.49$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 293.56$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 1474.57$
Joint optimization	$3.6435 \cdot 10^{10}$	$\boldsymbol{\delta}_9 = -3$ $\boldsymbol{\delta}_{12} = -2$ $\boldsymbol{\delta}_{17} = -4$ $\boldsymbol{\delta}_{23} = -2$ $\boldsymbol{\delta}_{24} = -2$ $\boldsymbol{\delta}_{25} = -1$ $\boldsymbol{\delta}_{41} = 0$ $\boldsymbol{\delta}_{42} = 0$ $\boldsymbol{\delta}_{43} = 0$ $\boldsymbol{\delta}_{44} = 1$	$\boldsymbol{\theta}^{\mathrm{r}}_2 = 12.59$ $\boldsymbol{\theta}^{\mathrm{r}}_4 = 17.63$ $\boldsymbol{\theta}^{\mathrm{r}}_5 = 13.46$ $\boldsymbol{\theta}^{\mathrm{r}}_7 = 13.21$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,0} = 1.23$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,1} = 197.78$ $\boldsymbol{\theta}^{\mathrm{v}}_{32,2} = 761.87$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,0} = 1.34$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,1} = 323.92$ $\boldsymbol{\theta}^{\mathrm{v}}_{41,2} = 1273.11$

decisions than the iterative optimization and the bi-level optimization, and the total cost is also slightly lower than the one obtained by the other two frameworks. In Table 4.3, iterative optimization, bi-level optimization and joint optimization yield the same topology design decisions, but different control parameters. However, the values of the total costs of these three solution frameworks are identical for the first five digits. Therefore, in this scenario, the topology design has a more dominant impact on the performance of the traffic network than the traffic control.

4.7 Conclusions

In this chapter, we have introduced a co-design method that jointly optimizes the network topology and traffic control parameters. We have proposed four different solution frameworks for such a co-design problem, namely separate optimization, iterative optimization, bi-level optimization, and joint optimization. One should choose the most appropriate solution framework according to the computational complexity requirements. We have tested these four solution frameworks in a simulation-based case study — the Singapore expressway network. The results show that joint optimization, bi-level optimization, and iterative optimization can result in a superior performance than separate optimization.

Part III

Chapter 5

Unmanned Aerial Vehicles for Monitoring Freeway Networks

In this chapter, Unmanned Aerial Vehicles (UAVs) are considered to monitor the traffic conditions in a freeway network, where the goal is to find an optimal path for each UAV. Two different monitoring settings are considered: (1) UAVs can monitor while flying, and (2) UAVs can only monitor when hovering in the air. The first setting is recast as a multiple rural postman problem. Next, the problem is translated into a multiple traveling salesman problem by mapping the freeway network into a virtual graph, and then the resulting problem is periodically solved using mixed-integer linear programming. The second setting is formulated as a Markov Decision Process (MDP). The freeway network is decomposed into multiple cells, and the task is to let UAVs visit the cells that cover the freeway links as frequently as possible. Since usually a freeway network is so large that the problem cannot be solved using standard MDP solution methods, three alternative solution methods are considered instead, namely fitted Q-iteration, model predictive control, and parametrized control. A simulation-based case study involving the Singapore expressway network is used to illustrate our approaches for the two settings.

5.1 Introduction

Traffic information is important for traffic management and control. In order to obtain real-time traffic information, surveillance systems are often used in freeway networks. Traditionally, fixed sensors are installed on roads, such as cameras or road-embedded sensors, and they cannot be moved once installed. Determining good sensor locations for flow observation and estimation are reviewed in [62]. However, fixed sensors cannot be deployed everywhere in the network due to limited budgets, and as a result, the available fixed sensors may fail to gather reliable representative traffic information in case of a dynamic traffic situation. This is the motivation to introduce mobile sensors in the traffic network.

Some work considers mobile sensors as probe vehicles, e.g., using floating car data (FCD) technique, which move in the same way as regular vehicles on the roads. FCD is can be used to estimate the traffic speed in the traffic network [82], based on the collection of localization data, such as vehicle speeds, vehicle headings and time information using mobile phones or on-board GPS devices. In contrast to conventional methods, no additional hardware installed in the traffic network is necessary. However, the reliability of travel time estimation based on FCD highly depends on the percentage of FCD-equipped cars that participate in

the traffic flow [54]. If the FCD-equipped cars are not enough in the traffic flow, it may not collect reliable traffic information.

Alternatively, Unmanned Aerial Vehicles (UAVs) can be chosen as mobile sensors for traffic surveillance. This approach was used as early as in 1965, when a transportation consultant in Maryland used a fixed-wing aircraft to collect traffic information [77]. In recent years, research studies have been done at many different universities and research institutes. The University of Florida initiated a project named Airborne Traffic Surveillance Systems [125], which mainly aims at monitoring remote and rural areas of the state of Florida by using UAVs. The UAV they used is called the Aerosonde UAV; it can endure over 32 hours at an altitude between 100 and 6000 meters above the ground. Linköping University is conducting a long-term fundamental research project on UAVs [42], in cooperation with a number of universities in Europe, USA, and South America. The goal is to develop technologies for a fully autonomous UAV operating over diverse geographical terrain containing road and traffic networks. Some other research activities include the experiments conducted by Ohio State University [34], the COMETS project funded by European Commission [60], and the Ultimate Auto-Pilot system built by University of California, Berkeley [57]. More information about this topic can be found in [117].

The advantage of using UAVs is that they can move anywhere above the traffic network, not directly being influenced by the traffic situation, and not limited by physical restrictions of the network, e.g., narrow roads or uneven terrain. In reality, each UAV usually has a limited local field of view around itself, and cannot cover the whole domain of interest all the time. Therefore, path planning strategies for the UAVs are crucial for a successful surveillance system, especially when monitoring a large-scale traffic network with a limited number of UAVs. In this way, all UAVs thus have to fly around to update the monitored traffic information.

It has to be emphasized that in the literature, many papers, e.g. [2, 20, 31, 143], use the term "path planning" to refer to finding a shortest or fastest path for each UAV from an initial point to a terminal point in the network. However, in this chapter, "path planning" means to find a path for each UAV so that the trajectories of all the UAVs cover the whole network or a selected part of it as well as possible w.r.t. some cost criteria.

We consider two different settings of UAVs monitoring the freeway network in this chapter:

- **Setting I**: Each UAV can monitor when flying along a link; so it has two modes — monitoring and traversing. If a UAV is monitoring a link, it flies with a low speed; if a UAV is traversing a link, it flies with a high speed. This setting is often used to track objects moving on the ground [132, 134].

- **Setting II**: Each UAV can only monitor when hovering in the air. Therefore, it only has one flying mode, and flies with a high speed. This setting is often used to inspect fixed targets, e.g., parking lots or buildings [34, 104].

In the first setting, the UAVs path planning problem can be generalized as finding a least-cost tour[1] on a specified set of arcs in a graph. It is closely related to two types of arc routing problems — the Chinese postman problem and the rural postman problem. The Chinese postman problem has been posed by Kwan [88]. Generally speaking, the goal is to seek a minimum-cost closed tour that visits all arcs of a graph. Hardgrave and Nemhauser [69]

[1] In this chapter, a tour refers to a path from an origin to a destination for a UAV.

have shown that this problem can be immediately transformed into a traveling salesman problem [90], and then solved by dedicated traveling salesman algorithms. However, more direct approaches are also possible (see e.g. [100]). A comprehensive literature on this topic can be found in [51]. The rural postman problem is to find a closed tour that visits each arc of a given subset on a graph at least once, with the total cost minimized. This type of problem usually underlies applications in practical contexts where it is not necessary to service all links of a network, such as street sweeping [17], garbage collection [10], mail delivery [91], and so on. Interested readers are referred to [52] for detailed information. The problem in this setting is a variant of the rural postman problem, including two types of links for UAV: the physical roads that are going to be monitored (each of them must be visited once and only once), and the aerial links that are just used for traversal (they can be visited if needed). The problem will be translated into a multiple traveling salesman problem, and solved by a mixed-integer linear programming (MILP) approach.

In the second setting, the UAVs path planning problem can be considered as a decision making process, in which the freeway network is divided into multiple cells. At each step, the decision maker chooses for each UAV which cell to move to, and then the UAV flies to the center of the chosen cell to monitor, e.g., by taking a picture. According to the monitoring performance, a reward is given to the decision maker so that he knows how good the chosen action was. This process can be formulated as a Markov Decision Process (MDP) [8]. However, the path planning problem for UAVs in a large-scale freeway network will result in an MDP problem with a big state space, which cannot be solved by standard MDP solution methods. In order to tackle this issue, we use three alternative solution methods to solve the problem in practice, namely fitted Q-iteration, model predictive control, and parameterized control.

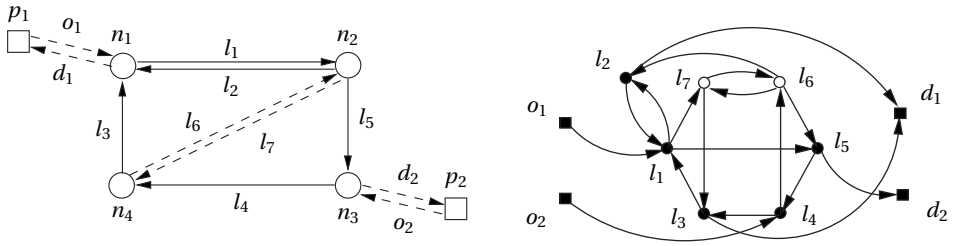
The rest of the chapter is organized as follows. Section 5.2 introduces the MILP approach to solve the UAVs path planning problem of the first setting. In Section 5.3, the problem of the second setting is formulated by using MDPs, and then solved. After that, simulations and results of a case study are shown in Section 5.4. Finally, conclusions are given in Section 5.5.

5.2 Setting I: Mixed-integer linear programming

5.2.1 Problem definition

The objective in this setting is to let UAVs *monitor* a selected subset of physical links in the freeway network during each monitoring period $[\kappa T_{\mathrm{m}}, (\kappa+1)T_{\mathrm{m}}]$ for $\kappa = 0, 1, 2, \ldots$, with T_{m} the length of the monitoring period. Note that the value of T_{m} should be selected large enough to allow the UAVs to finish monitoring all the selected links. The basic idea we propose is to translate the freeway network into a virtual graph (see Figure 5.1 for an example), mapping each selected link to a vertex, with the relationship $l = \ell(i)$ indicating that link l corresponds to vertex i. In order to avoid confusion, in this chapter, we use the terms “links" and “nodes" to refer to the freeway network, and “arcs" and “vertices" to refer to the virtual graph. The links to be monitored are selected based on one or more of the following conditions:

1. The last monitoring action to link $\ell(i)$ has happened a long time ago, or link $\ell(i)$ was not even monitored yet;



(a) Freeway network (b) Virtual graph

Figure 5.1: Example: mapping a freeway network into a virtual graph. The physical links l_1-l_5 are mapped to vertices indicated by black dots, the pure aerial links l_6 and l_7 are mapped to vertices indicated by circles, and the incoming and outgoing links of the depots p_1 and p_2 are mapped to vertices indicated by black squares.

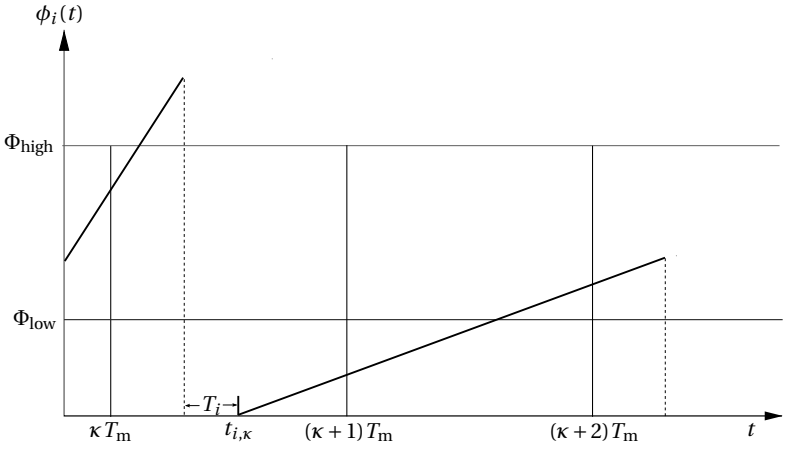


Figure 5.2: Example of the evolution of attraction level in different periods: link $\ell(i)$ should be monitored at most once during the period $[\kappa T_{\text{m}}, (\kappa+1)T_{\text{m}}]$ because $\Phi_{low} \leqslant \phi_i^{\text{d}}(\kappa) < \Phi_{high}$, and $\ell(i)$ should not be monitored during the period $[(\kappa+1)T_{\text{m}}, (\kappa+2)T_{\text{m}}]$ because $\phi_i^{\text{d}}(\kappa+1) < \Phi_{low}$, with $t_{i,\kappa}$ the time instant at which a UAV leaves $\ell(i)$ after monitoring the link during the κth monitoring period, and T_i the time spent by a UAV on link $\ell(i)$.

2. The traffic density[2] of link $\ell(i)$ is high;

3. Link $\ell(i)$ is so vital that it needs frequent monitoring.

All these conditions can be captured by defining a variable called attraction level $\phi_i(t)$. In general, the larger $\phi_i(t)$ is, the higher the probability will be that link $\ell(i)$ has to be selected for monitoring in the upcoming period. The attraction level $\phi_i(t)$ is assumed to increase as time elapses, until link $\ell(i)$ is monitored by a UAV, and then $\phi_i(t)$ is reset to zero. In this way, the first condition is guaranteed. We let $\phi_i(t)$ increase with a piecewise constant slope, where the slope can change after each monitoring action and at the beginning of each monitoring period (see e.g. Figure 5.2). Moreover, the rate of increase of the attraction level does not only depend on the traffic density of link $\ell(i)$, but it can also be manually tuned by the operators in the traffic control centers. Thus, the second and the third conditions are guaranteed, too. Below we will show that by introducing the attraction level the dynamic nature of the freeway network can be captured within a mixed-integer linear programming (MILP) approach; in particular, a rural postman problem has to be solved periodically. In this way, this problem can be considered as a *periodical* multiple rural postman problem, by mapping the freeway network into a virtual graph, as will be explained next.

All important symbols used in this section are listed in Table 5.1.

Freeway network

The freeway network contains a set $\mathscr{L}_{\text{phys}}$ of physical links that represent roads to be monitored, a set $\mathscr{L}_{\text{air}}$ of aerial links that are only used for traversing by UAVs, and a set $\mathscr{D}$ of depots where UAVs start and finish their tours. Without loss of generality, it is assumed that each depot only has one incoming link and one outgoing link. In each depot p, there are N_p UAVs that can be used to monitor the traffic network, with C_p the capacity of depot p ($N_p \leqslant C_p$). The total number of UAVs is thus $N = \sum_{p\in\mathscr{D}} N_p$. Note that the physical links can be visited not only for monitoring purposes, but also for traversing purposes, just as the aerial links. We assume that a UAV is able to accurately monitor the traffic situation on a link while flying across it at a low speed. So, if a UAV monitors a link, the link should be visited only *once*, and the monitoring speed of a UAV is v_{low}; if a UAV traverses a link, the link can be visited multiple times, and the traversing speed of a UAV is v_{high}.

The subset $\mathscr{L}_{\text{phys,select}}(\kappa)$ of physical links to be monitored is selected at the beginning of each period. The attraction level $\phi_i(t)$ is sampled at each time instant $t = \kappa T_{\text{m}}$ for $\kappa = 0, 1, \ldots$, with $\phi_i^{\text{d}}(\kappa) = \phi(\kappa T_{\text{m}})$. Two thresholds Φ_{low} and Φ_{high} are defined: if $\Phi_{\text{low}} \leqslant \phi_i^{\text{d}}(\kappa) < \Phi_{\text{high}}$, then $\ell(i)$ is considered to be "moderately attractive", which means that a UAV may monitor it or not during the κth period; if $\phi_i^{\text{d}}(\kappa) \geqslant \Phi_{\text{high}}$, then $\ell(i)$ is considered to be "highly attractive", which means a UAV must monitor it during the κth period.

Virtual graph

The virtual graph[3] $\mathscr{G} = (\mathscr{V}, \mathscr{A})$ is defined according to the given freeway network, where $\mathscr{V}$ is the set of vertices associated with the links in the freeway network, and $\mathscr{A} = \in \mathscr{V} \times \mathscr{V}\}$ is the set of arcs connecting the vertices. The sets $\mathscr{V}$ and $\mathscr{A}$ are constructed as follows.

[2] If a traffic flow model is available, the traffic density is predicted for the upcoming monitoring period; otherwise, it is based on measurement in the previous periods.

[3] In fact, the graph $\mathscr{G}$ may be different for each period $[\kappa T_{\text{m}}, (\kappa+1) T_{\text{m}}]$, but for the sake of compactness, the index κ of $\mathscr{G}$ is omitted in the chapter.

Table 5.1: Symbols used for Setting I

symbol	meaning
$\mathscr{L}_{\text{phys}}$	set of physical links
$\mathscr{L}_{\text{air}}$	set of aerial links
$\mathscr{D}$	set of depots
N_p	number of UAVs at depot p
C_p	capacity of depot p
$\mathscr{L}_{\text{phys,select}}(\kappa)$	set of selected physical links to be monitored during the κth period
$\mathscr{V}_{\text{moni}}$	set of vertices corresponding to the physical links to be monitored
$\mathscr{V}_{\text{moni,high}}$	set of vertices corresponding to the physical links that must be monitored once and only once
$\mathscr{V}_{\text{moni,mod}}$	set of vertices corresponding to the physical links that should be monitored not more than once
$\mathscr{V}_{\text{trav,base}}$	set of vertices corresponding to both the physical links and the aerial links
$\mathscr{V}_{\text{trav}}$	set of vertices corresponding to the links used for traversing
$\mathscr{V}_{\text{orig}}$	set of origins
$\mathscr{V}_{\text{dest}}$	set of destinations
$\mathscr{Q}_i$	set of duplications of vertex i
$\mathscr{I}_i$	set of incoming arcs of vertex i
$\mathscr{O}_i$	set of outgoing arcs of vertex i
$\mathscr{N}$	set of UAVs
T_{m}	length of the monitoring period
$\phi_i(t)$	attraction level on link $\ell(i)$
$\phi_i^{\text{d}}(\kappa)$	sampled attraction level at time instant $t = \kappa T_{\text{m}}$
Φ_{low}, Φ_{high}	thresholds for the attraction levels
$a_i(\kappa)$	rate of increase of the attraction level for the κth period
$\bar{\rho}_{i,\kappa}$	average density measured on link $\ell(i)$ during the κth period
$\rho_{\text{crit},i}$	critical density of link $\ell(i)$
α_i, β_i, γ_i	parameters for computing $a_i(\kappa)$
$\tau_{i,n}(\kappa)$	arrival time of UAV n at link $\ell(i)$ during the κth period
$e_{i,n}(\kappa)$	energy level of UAV n when entering link $\ell(i)$ during the κth period
$T_{i,n}$	time spent by UAV n on link $\ell(i)$
$E_{i,n}$	energy consumed by UAV n on link $\ell(i)$
$v_{i,n}$	speed of UAV n on link $\ell(i)$
L_i	length of link $\ell(i)$
$C_{\text{a},n}$	activation cost of UAV n
$c_i(\kappa)$	auxiliary variable for vertex i for cycle imposement constraint during the κth period
$E_n^{\max}$	maximum energy level of UAV n
M_t, M_e	large positive constants

- The vertex set consists of three groups:

  - The first group (denoted by the set $\mathcal{V}_{\text{moni}}$) represents the physical links that should be monitored by UAVs in the freeway network, with $\mathcal{V}_{\text{moni}} = \{i | \ell(i) \in \mathcal{L}_{\text{phys,select}}(\kappa)\}$. Moreover, this group can be further divided into two subgroups: $\mathcal{V}_{\text{moni,high}}$ which contains the vertices that corresponds to the "highly attractive" links, which must be monitored once and only once, and $\mathcal{V}_{\text{moni,mod}}$ which contains the vertices that corresponds to the "moderately attractive" links, which should be monitored not more than once.

  - The second group (denoted by the set $\mathcal{V}_{\text{trav}}$) represents the links that are used by the UAVs for traversing. Any link $\ell(i)$ in the freeway network can be traversed multiple times. However, since we will use a binary optimization variable to indicate whether or not a link is visited by a given UAV, it is unknown how many times a link is traversed by a given UAV. In order to tackle this issue, each vertex $i \in \mathcal{V}_{\text{trav,base}}$ is duplicated Q_i times, with $\mathcal{V}_{\text{trav,base}} = \{i | \ell(i) \in \mathcal{L}_{\text{phys}} \cup \mathcal{L}_{\text{air}}\}$, and Q_i a constant. The qth duplication $i^{[q]}$ of vertex i indicates that link $\ell(i)$ may be traversed for the qth time by the same UAV. In this way, the set $\mathcal{V}_{\text{trav}}$ is defined as:

$$\mathcal{V}_{\text{trav}} = \bigcup_{i \in \mathcal{V}_{\text{trav,base}}} \mathcal{Q}_i \tag{5.1}$$

    with $\mathcal{Q}_i = \{i^{[1]}, i^{[2]}, \ldots, i^{[Q_i]}\}$. So if a *physical* link $\ell(i)$ is visited by a UAV for the first time, link $\ell(i)$ is considered to be monitored, and thus vertex $i \in \mathcal{V}_{\text{moni}}$ is visited; if link $\ell(i)$ is visited for the second time, and so on, then link $\ell(i)$ is considered to be traversed, and vertices $i^{[1]}, i^{[2]}, \ldots \in \mathcal{V}_{\text{trav}}$ are visited.

  - The third group of vertices includes the set $\mathcal{V}_{\text{orig}}$ of origins, and the set $\mathcal{V}_{\text{dest}}$ of destinations, which contains the vertices that corresponds to the outgoing and incoming links of depots respectively. Moreover, if vertex $o \in \mathcal{V}_{\text{orig}}$ corresponds to the outgoing link of depot p, and vertex $d \in \mathcal{V}_{\text{dest}}$ corresponds to the incoming link of depot p, the relationship between d and o is defined as $d = \text{dest}(o)$. All the UAVs are initially put at origins, and the set of all the UAVs is

$$\mathcal{N} = \bigcup_{o \in \mathcal{V}_{\text{orig}}} \mathcal{N}_o \tag{5.2}$$

    with $\mathcal{N}_o$ the set of UAVs at origin o.

  As a result, the vertex set equals $\mathcal{V} = \mathcal{V}_{\text{moni}} \cup \mathcal{V}_{\text{trav}} \cup \mathcal{V}_{\text{orig}} \cup \mathcal{V}_{\text{dest}}$.

- The arc set represents the physical restrictions of link connections in the freeway network. For each vertex $i \in \mathcal{V}$, an incoming neighborhood $\mathcal{I}_i$ is defined as the set of vertices that are directly connected to vertex i by one of its incoming arcs. Similarly, $\mathcal{O}_i$ is defined as the outgoing neighborhood of vertex i. Moreover, a binary-valued variable $x_{i,j,n}(\kappa)$ is associated with each arc (i, j), indicating whether $(x_{i,j,n}(\kappa) = 1)$ or not $(x_{i,j,n}(\kappa) = 0)$ vertex j is visited directly after vertex i by UAV n during the κth period $[\kappa T_{\text{m}}, (\kappa + 1) T_{\text{m}}]$.

5.2.2 Mathematical formulation

Attraction level update

The attraction level is only calculated at each time step $\kappa = 0, 1, \ldots$, increasing with a constant rate $a_i(\kappa)$. The initial attraction level $\phi_i^{\mathrm{d}}(0)$ and the initial rate $a_i(0)$ can be computed based on a standard value for all the links, or based on historical data for each individual link. There are two cases for updating the attraction level at the end of the κth period $[\kappa T_{\mathrm{m}}, (\kappa+1)T_{\mathrm{m}}]$:

1. If link $\ell(i)$ is monitored by a UAV during the κth period, the attraction level is computed as

$$\phi_i^{\mathrm{d}}(\kappa+1) = a_i(\kappa)((\kappa+1)T_{\mathrm{m}} - t_{i,\kappa}) \tag{5.3}$$

where $t_{i,\kappa}$ denotes the time instant at which the UAV leaves link $\ell(i)$ after monitoring it during the κth period, calculated as

$$t_{i,\kappa} = \sum_{j \in \bar{\mathcal{O}}_i} \sum_{n \in \mathcal{N}} (\tau_{i,n}(\kappa) + T_{i,n}) x_{i,j,n}(\kappa), \tag{5.4}$$

with $\tau_{i,n}(\kappa)$ the arrival time of UAV n at link $\ell(i)$ during the κth period (see (5.31)), and $T_{i,n}$ the time spent by UAV n monitoring link $\ell(i)$ for $i \in \mathcal{V}_{\mathrm{moni}}$ (see (5.12)). The value of the rate $a_i(\kappa)$ depends on the density measured on link $\ell(i)$. More specifically, if the average density $\bar{\rho}_{i,\kappa}$ of link $\ell(i)$ during the κth period is larger than the critical density $\rho_{\mathrm{crit},i}$, the attraction level will increase at a higher rate. Therefore, the rate $a_i(\kappa)$ can be computed e.g. by:

$$a_i(\kappa) = \max\left(\alpha_i, \alpha_i e^{\left(\beta_i\left(\frac{\bar{\rho}_{i,\kappa}}{\gamma_i \rho_{\mathrm{crit},i}} - 1\right)\right)}\right) \tag{5.5}$$

with $\alpha_i, \beta_i > 0$ constants reflecting the importance of link $\ell(i)$, and $0 < \gamma_i \leqslant 1$ a threshold factor for the critical density.

2. If link $\ell(i)$ is not monitored during the κth period, the attraction level is computed as

$$\phi_i^{\mathrm{d}}(\kappa+1) = \phi_i^{\mathrm{d}}(\kappa) + a_i(\kappa) T_{\mathrm{m}} \tag{5.6}$$

and the rate of increase is computed as

$$a_i(\kappa) = a_i(\kappa - 1) \tag{5.7}$$

Remark: If a traffic flow model is available, then instead of using (5.7) the rate $a_i(\kappa)$ can be calculated based on the estimated instantaneous density $\hat{\rho}_i(\kappa T_{\mathrm{m}})$ by using (5.5).

Objective function

The objective function used in this chapter establishes a trade-off between minimizing the total travel time, the total energy consumption, and the total activation cost of the UAVs, and maximizing the total attraction levels on the monitored links during each monitoring period. The trade-off can be made among the different objectives by using weighting coefficients

$\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5 \geqslant 0$ in the objective function:

$$J_{\text{MILP}}(\kappa) = \lambda_1 J_{\text{energy}}(\kappa) + \lambda_2 J_{\text{time}}(\kappa) + \lambda_3 J_{\text{active}}(\kappa) + \lambda_4 J_{\text{attract}}(\kappa) + \lambda_5 J_{\text{num}}(\kappa) \tag{5.8}$$

The definitions of $J_{\text{energy}}(\kappa)$, $J_{\text{time}}(\kappa)$, $J_{\text{active}}(\kappa)$, $J_{\text{attract}}(\kappa)$, and $J_{\text{num}}(\kappa)$ are given next.

- *Energy consumption*: In this chapter, the energy consumption is assumed to be related with the speed of the UAVs, and to be proportional to the length of the links. The energy consumed when link $\ell(i)$ is visited is then formulated as

$$E_{i,n} = f_{\text{e}}(v_{i,n}) \cdot L_i, \quad \forall i \in \mathcal{V} \tag{5.9}$$

  where f_{e} represents the relationship between the speed of the UAV and the energy consumption per unit distance, L_i is the length of link $\ell(i)$, and $v_{i,n}$ denotes the speed of a UAV n on link $\ell(i)$, with

$$v_{i,n} = \begin{cases} v_{\text{low}}, & \text{if } i \in \mathcal{V}_{\text{moni}} \\ v_{\text{high}}, & \text{else} \end{cases} \tag{5.10}$$

  The total energy consumption is then

$$J_{\text{energy}}(\kappa) = \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{O}_i} \sum_{n \in \mathcal{N}} E_{i,n} \cdot x_{i,j,n}(\kappa) \tag{5.11}$$

- *Travel time*: This depends on the length of each link in the freeway network, and the speed of the UAV when visiting that link. The time spent on link $\ell(i)$ by UAV n is formulated as

$$T_{i,n} = \frac{L_i}{v_{i,n}} \ . \tag{5.12}$$

  Note that both $T_{i,n}$ and $E_{i,n}$ are constants so that our problem is ensured to be an MILP problem. The total travel time is formulated similarly to the total energy consumption:

$$J_{\text{time}}(\kappa) = \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{O}_i} \sum_{n \in \mathcal{N}} T_{i,n} \cdot x_{i,j,n}(\kappa) \tag{5.13}$$

- *Fixed activation cost*: An activation cost $C_{\text{a},n}$ is incurred whenever a UAV n is activated. The total cost of using UAVs is then:

$$J_{\text{active}}(\kappa) = \sum_{o \in \mathcal{V}_{\text{orig}}} \sum_{i \in \mathcal{O}_o} \sum_{n \in \mathcal{N}} C_{\text{a},n} \cdot x_{o,i,n}(\kappa) \tag{5.14}$$

- *Attraction levels*: Since each vertex $i \in \mathcal{V}_{\text{moni,high}}$ must be visited exactly once, the sum of the attraction levels on the corresponding links is fixed. Therefore, only the links that correspond to the vertices in $\mathcal{V}_{\text{moni,mod}}$ should be considered in the optimization problem. The goal is to maximize the sum of the attraction levels on all the monitored links, which is equivalent to minimizing the sum of the attraction levels on all the non-

monitored links. Therefore, the objective function is formulated as:

$$J_{\text{attract}}(\kappa) = \sum_{i \in \mathcal{V}_{\text{moni,mod}}} \phi_i^{\text{d}}(\kappa) \left(1 - \sum_{j \in \mathcal{O}_i} \sum_{n \in \mathcal{N}} x_{i,j,n}(\kappa)\right) \tag{5.15}$$

with the term $\sum_{j \in \mathcal{O}_i} \sum_{n \in \mathcal{N}} x_{i,j,n}(\kappa)$ indicating whether link $\ell(i)$ is monitored or not.

- *Number of monitoring actions*: For the links that correspond to the vertices in $\mathcal{V}_{\text{moni,mod}}$, it could happen that the attraction level on one link equals the sum of the attraction levels on several other links. In this extreme case, UAVs should be encouraged to monitor multiple links instead of that single link with the highest attraction level only. In order to do so, the sum of the number of times that each link $\ell(i)$ for $i \in \mathcal{V}_{\text{moni,mod}}$ is visited is also considered:

$$J_{\text{num}}(\kappa) = \sum_{i \in \mathcal{V}_{\text{moni,mod}}} \left(1 - \sum_{j \in \mathcal{O}_i} \sum_{n \in \mathcal{N}} x_{i,j,n}(\kappa)\right) \tag{5.16}$$

Constraints

- *Assignment constraints*: For any origin $o \in \mathcal{V}_{\text{orig}}$, the UAVs can stay idle. This means that each arc $(o,i) \in \mathcal{A}$ with $o \in \mathcal{V}_{\text{orig}}$, $i \in \mathcal{O}_o$ can be visited by a UAV n at most once:

$$\sum_{i \in \mathcal{O}_o} x_{o,i,n}(\kappa) \leqslant 1, \quad \forall o \in \mathcal{V}_{\text{orig}}, \forall n \in \mathcal{N}_o \tag{5.17}$$

and a UAV n can only leave its own depot:

$$x_{o,i,n}(\kappa) = 0, \quad \forall i \in \mathcal{O}_o, \forall o \in \mathcal{V}_{\text{orig}}, \forall n \notin \mathcal{N}_o \tag{5.18}$$

When UAVs finish their tours and return to the depots, there are two different cases:

  - **Case A**: A UAV can return to any depot. In this case, the number of UAVs at each depot should not exceed the capacity of that depot (recall that the relationship $d = \text{dest}(o)$ represents origin o and destination d corresponding to the same depot):

$$N_o + \sum_{n \in \mathcal{N}} \sum_{i \in \mathcal{I}_{\text{dest}(o)}} x_{i,\text{dest}(o),n}(\kappa) - \sum_{n \in \mathcal{N}_o} \sum_{j \in \mathcal{O}_o} x_{o,j,n}(\kappa) \leqslant C_{\text{dest}(o)}, \quad \forall o \in \mathcal{V}_{\text{orig}} \tag{5.19}$$

    with N_o the number of UAVs at the depot that corresponds to origin o, and C_d the capacity of depot that corresponds to destination d. Moreover, the total number of outgoing UAVs equals the total number of returning UAVs:

$$\sum_{o \in \mathcal{V}_{\text{orig}}} \sum_{n \in \mathcal{N}_o} \sum_{j \in \mathcal{O}_o} x_{o,j,n}(\kappa) = \sum_{o \in \mathcal{V}_{\text{orig}}} \sum_{n \in \mathcal{N}_o} \sum_{i \in \mathcal{I}_{\text{dest}(o)}} x_{i,\text{dest}(o),n}(\kappa), \tag{5.20}$$

  - **Case B**: Each UAV must return to its original depot. In this case, the number of outgoing UAVs should be the same as the number of returning UAVs for each

depot:

$$\sum_{n\in\mathcal{N}_o}\sum_{j\in\mathcal{O}_o} x_{o,j,n}(\kappa) = \sum_{n\in\mathcal{N}_o}\sum_{i\in\mathcal{I}_{\text{dest}(o)}} x_{i,\text{dest}(o),n}(\kappa),\ \ \forall o\in\mathcal{V}_{\text{orig}} \tag{5.21}$$

Moreover, a so-called cycle imposement constraint, which has been introduced by [25], is added. Generally speaking, the cycle imposement method associates a unique current $c_o(\kappa)$ with each origin vertex $o\in\mathcal{V}_{\text{orig}}$, and ensures that $c_d(\kappa) = c_o(\kappa)$ if $d = \text{dest}(o)$, with $c_d(\kappa)$ the current at destination d. If a vertex j is preceded by another vertex i, these two vertices share the same current $c_i(\kappa) = c_j(\kappa)$. In this way, the tour is imposed to make the UAVs return to the same depot as the one they started from. The following constraints ensure this:

$$c_{o,n}(\kappa) = o,\ \ \forall o\in\mathcal{V}_{\text{orig}}, \forall n\in\mathcal{N}_o \tag{5.22}$$

$$c_{\text{dest}(o),n}(\kappa) = c_{o,n}(\kappa),\ \ \forall o\in\mathcal{V}_{\text{orig}}, \forall n\in\mathcal{N}_o \tag{5.23}$$

$$c_{i,n}(\kappa) - c_{j,n}(\kappa) \leqslant (D-1)(1-x_{i,j,n}(\kappa)-x_{j,i,n}(\kappa)),\ \ \forall i,j\in\mathcal{V} \tag{5.24}$$

with D the number of depots, and where it is assumed without loss of generality that $\mathcal{V}_{\text{orig}} = \{1,2,\ldots,D\}$.

A link $\ell(i)$ for $i\in\mathcal{V}_{\text{moni,high}}$ should be visited once and only once by any UAV, which means that each vertex $i\in\mathcal{V}_{\text{moni,high}}$ is succeeded and preceded by exactly one vertex in the route of the UAVs:

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{O}_i} x_{i,j,n}(\kappa) = 1,\ \ \forall i\in\mathcal{V}_{\text{moni,high}}, \tag{5.25}$$

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{I}_i} x_{j,i,n}(\kappa) = 1,\ \ \forall i\in\mathcal{V}_{\text{moni,high}}, \tag{5.26}$$

Moreover, a link $\ell(i)$ for $i\in\mathcal{V}_{\text{moni,mod}}$ should be visited no more than once by any UAV, which is formulated as:

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{O}_i} x_{i,j,n}(\kappa) \leqslant 1,\ \ \forall i\in\mathcal{V}_{\text{moni,mod}}, \tag{5.27}$$

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{I}_i} x_{j,i,n}(\kappa) \leqslant 1,\ \ \forall i\in\mathcal{V}_{\text{moni,mod}}, \tag{5.28}$$

For a vertex $i\in\mathcal{V}_{\text{moni}}\cup\mathcal{V}_{\text{trav}}$, the number of times that UAV n enters link $\ell(i)$ should equal the number of times that UAV n exits link $\ell(i)$:

$$\sum_{j'\in\mathcal{I}_i} x_{j',i,n}(\kappa) = \sum_{j\in\mathcal{O}_i} x_{i,j,n}(\kappa),\ \ \forall i\in\mathcal{V}_{\text{moni}}\cup\mathcal{V}_{\text{trav}},\ \ \forall n\in\mathcal{N} \tag{5.29}$$

- *Travel time constraints*: As it is known from the traveling salesman literature [6], the assignment constraints mentioned above do not avoid subtours in the network, which means that tours could be formed between vertices in $\mathcal{V}_{\text{moni}}\cup\mathcal{V}_{\text{trav}}$ only and not be connected to any depot. One way to tackle this issue is to use so-called cycle elimination constraints. The idea behind these constraints is to assign an additional variable u_i to each vertex i, representing a vertex voltage. These vertex voltages have bounded values, and they increase at each vertex along the route until a terminal depot is reached.

By using these constraints, if a subtour exists, then no terminal depot is included there, so the voltages at the vertices in this route will increase to infinity, which is a contradiction.

One of the well-known approaches is called the Miller-Tucker-Zemlin cycle elimination constraints approach, which was introduced by Miller et al. [101]. We follow this method in this chapter. However, instead of using the vertex voltages, which have no practical meaning in our application, we introduce an arrival time variable $\tau_{i,n}(\kappa)$, which is the time instant that UAV n arrives at link $\ell(i)$, to act as the vertex voltage. For each UAV at an origin, that time is defined as:

$$\tau_{o,n}(\kappa) = \kappa T_{\mathrm{m}}, \quad \forall o \in \mathcal{V}_{\mathrm{orig}}, \forall n \in \mathcal{N} \tag{5.30}$$

If a vertex j succeeds a vertex i for the same UAV, the arrival time $\tau_{j,n}(\kappa)$ is equal to the arrival time $\tau_{i,n}(\kappa)$, plus the travel time spent by UAV n on vertex i:

$$\tau_{j,n}(\kappa) = \tau_{i,n}(\kappa) + T_{i,n}, \quad \text{if } x_{i,j,n}(\kappa) = 1, \tag{5.31}$$

for all $i, j \in \mathcal{V}_{\mathrm{moni}} \cup \mathcal{V}_{\mathrm{trav}}$. This equation is equivalent to

$$(\tau_{i,n}(\kappa) - \tau_{j,n}(\kappa) + T_{i,n}) x_{i,j,n}(\kappa) = 0, \quad \forall i, j \in \mathcal{V}_{\mathrm{moni}} \cup \mathcal{V}_{\mathrm{trav}}, \forall n \in \mathcal{N} \tag{5.32}$$

and it can be redefined as a linear inequality constraint by using the big-M method [131]

$$\tau_{i,n}(\kappa) - \tau_{j,n}(\kappa) + T_{i,n} + M_{\mathrm{t}} x_{i,j,n}(\kappa) \leqslant M_{\mathrm{t}}, \tag{5.33}$$

$$\tau_{j,n}(\kappa) - \tau_{i,n}(\kappa) - T_{i,n} + M_{\mathrm{t}} x_{i,j,n}(\kappa) \leqslant M_{\mathrm{t}}, \tag{5.34}$$

with M_{t} a large positive constant.

- *Energy level constraints*: The energy level constraints guarantee that all UAVs return to the depots before their batteries are depleted or before they run of fuel. Therefore, each UAV should never run out of energy:

$$0 \leqslant e_{i,n}(\kappa) \leqslant E_n^{\max}, \quad \forall i \in \mathcal{V}, \forall n \in \mathcal{N} \tag{5.35}$$

Moreover, similar to the travel time constraints, the energy level is defined as

$$e_{o,n}(\kappa) = E_n^{\max}, \quad \forall o \in \mathcal{V}_{\mathrm{orig}}, \forall n \in \mathcal{N} \tag{5.36}$$

$$(e_{i,n}(\kappa) - e_{j,n}(\kappa) - E_{i,n}) x_{i,j,n}(\kappa) = 0, \quad \forall i, j \in \mathcal{V}, \forall n \in \mathcal{N} \tag{5.37}$$

with $E_{i,n}$ computed by (5.9), and $E_n^{\max}$ the maximum energy level that UAV n can have. Similarly, (5.37) can be recast as linear constraints using the big-M method

$$e_{i,n}(\kappa) - e_{j,n}(\kappa) - E_{i,n} + M_{\mathrm{e}} x_{i,j,n}(\kappa) \leqslant M_{\mathrm{e}}, \tag{5.38}$$

$$e_{j,n}(\kappa) - e_{i,n}(\kappa) + E_{i,n} + M_{\mathrm{e}} x_{i,j,n}(\kappa) \leqslant M_{\mathrm{e}}, \tag{5.39}$$

with M_{e} a large positive constant.

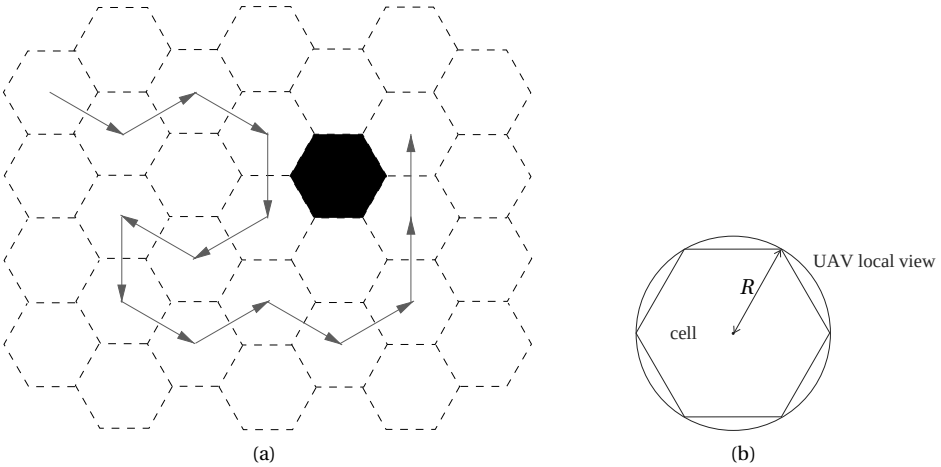


Figure 5.3: (a) Network is decomposed into cells, with the black cell indicating that no freeway link is present in that cell, and UAVs only visit the cells containing freeway links; (b) Relationship between the local field of view of the UAVs and the hexagonal cell.

5.2.3 Solution methods

The optimization problem for each period $[\kappa T_{\mathrm{m}}, (\kappa+1)T_{\mathrm{m}}]$ is formulated as:

$$\begin{aligned} \min_{x_{i,j,n}(\kappa),\, \tau_{i,n}(\kappa),\, e_{i,n}(\kappa),\, c_{i,n}(\kappa)} \quad & J_{\mathrm{MILP}}(\kappa) \\ \text{subject to} \quad & \Phi_{\mathrm{MILP}}(x_{i,j,n}(\kappa),\, \tau_{i,n}(\kappa),\, e_{i,n}(\kappa),\, c_{i,n}(\kappa)) = 0 \\ & \Psi_{\mathrm{MILP}}(x_{i,j,n}(\kappa),\, \tau_{i,n}(\kappa),\, e_{i,n}(\kappa),\, c_{i,n}(\kappa)) \leqslant 0 \end{aligned} \tag{5.40}$$

where $\Phi_{\mathrm{MILP}}(\cdot)$ captures all the equality constraints (5.18), (5.20)-(5.23), (5.27)-(5.30), and (5.36), and $\Psi_{\mathrm{MILP}}(\cdot)$ captures all the inequality constraints (5.17), (5.19), (5.24), (5.33)-(5.35), (5.38), and (5.39).

This is a standard mixed-integer linear programming program. It can be solved using cutting plane methods [81], or branch and bound methods [89].

5.3 Setting II: Markov decision processes

5.3.1 Problem definition

Since each UAV only has a limited local field of view around itself, the traffic network is accordingly decomposed into small cells, as shown in Figure 5.3(a). If a cell covers no freeway link, that cell is called an empty cell. The task of the UAVs is to consecutively move and monitor each non-empty cell. The location of each cell is characterized by its center, denoted by y_i, with $i \in \{1,2,\ldots,N_{\mathrm{c}}\}$, with N_{c} the total number of non-empty cells. In this chapter, a regular hexagon is chosen as the shape of the cell. The advantage of using this hexagonal decomposition is that the distance between the centers of any two adjacent cells is identical. If each UAV is considered as a point mass moving from the center of one cell to the center

of another at each time step, only the heading of the UAVs is controlled, while the speed is considered as a constant v_{high}. The size of the hexagonal cell depends on the size of the local view of the UAVs. We assume that a UAV has a circular local view with a radius R around its position. Then, the distance between the center and any vertex of the hexagon is equal to R, as shown in Figure 5.3(b). Therefore, the length of each time step is[4] $T = L/v_{\text{high}}$, with L is the distance between the centers of two adjacent cells: $L = \sqrt{3}R$.

The objective of an MDP problem is to maximize the return, i.e. the cumulative aggregation of rewards along a trajectory starting from the initial state. Therefore, the reward in this setting should be defined w.r.t. the monitoring performance. The uncertainty about the traffic situation in each cell keeps increasing until that cell is visited by a UAV, which means that the monitoring performance depends on how frequently the traffic information for each cell is updated. Therefore, the UAVs should move to the cells that have not been visited for a long time. In this way, the current state (e.g. the positions of the UAVs) will depend on the previous states. However, an MDP problem requires that the process possesses the Markov property, i.e., the next state of the process and the given reward only depend on the current state of the process and the current action of the decision maker, and they are (conditionally) independent of all previous states and actions, which is a contradiction.

In order to guarantee the Markov property of our problem, an interest level $\eta_i(k)$ at each cell i is defined as an auxiliary state variable. If any UAV visits a cell i at time step k, the interest level $\eta_i(k)$ is immediately set to a minimum value $\eta_{\min,i}$ so that cell i will temporarily not be visited by other UAVs; otherwise, $\eta_i(k)$ will keep increasing at each time step k. The higher $\eta_i(k)$ is, the more likely a UAV will visit cell i. The advantage of introducing the interest level is threefold:

1. As a state variable, the interest level in the current step only depends on the one in the previous step, just as the other state variable, i.e., the current position of each UAV, so that the Markov property of the problem is guaranteed;
2. The interest level can be directly considered as the reward to be granted to a UAV, if it visits the corresponding cell;
3. The interest level is used as a indirect medium of communications that allows UAVs to coordinate their actions.

Remark: In fact, the interest level and the attraction level defined in setting I are both a measure of need to visit a cell/link.

All symbols used in this section are listed in Table 5.2.

5.3.2 Mathematical formulation

UAV state transition function

Let $p_n(k) \in \mathbb{R}^2$ denote the position of the nth UAV at time step k, for $k = 0, 1, \ldots$, and $n = 1, 2, \ldots, N_{\text{u}}$, with N_{u} the total number of UAVs. The state transition function for the UAV position is

$$p_n(k+1) = p_n(k) + L \begin{bmatrix} \cos\theta_n(k) \\ \sin\theta_n(k) \end{bmatrix}, \tag{5.41}$$

[4] In general, the value of T is much smaller than the value of the length of the monitoring period T_{m} in Setting I.

Table 5.2: Symbols used for Setting II

symbol	meaning
$p_n(k)$	position of UAV n at time step k
$\theta_n(k)$	controllable heading of UAV n at time step k
$\eta_i(k)$	interest level of cell i at time step k
$\eta_{\min,i}$	minimum interest level of cell i
N_u	number of UAVs
N_c	number of non-empty cells
N_i	number of interest levels after discretization
T	length of time step
R	radius of cell

with $p_n(k) \in \{y_i | i = 1,2,\ldots,N_\mathrm{c}\}$, and $\theta_n(k) \in \Theta = \{\frac{\pi}{6}, \frac{\pi}{2}, \frac{5\pi}{6}, \frac{7\pi}{6}, \frac{3\pi}{2}, \frac{11\pi}{6}\}$ the controllable heading of UAV n at time step k.

Moreover, the interest level $\eta_i(k)$ at each cell i is a state variable as well. An advanced way to calculate the interest level is to take the traffic dynamics into account, e.g., if traffic flows in a cell become more dense, that cell should get a higher interest level. The state transition function is thus generally formulated as:

$$\eta_i(k+1) = \begin{cases} F_\mathrm{reset}(\eta_{\min,i}, \hat{\rho}_i(k)), & \text{if } p_n(k) = y_i \\ F_\mathrm{update}(\eta_i(k), \hat{\rho}_i(k)), & \text{else} \end{cases} \tag{5.42}$$

with $\eta_{\min,i}$ the minimum interest level of cell i, and $\hat{\rho}_i(k)$ the estimated traffic density in cell i at time step k. One possible way to formulate F_reset and F_update is similar to (5.5):

$$F_\mathrm{reset}(\eta_{\min,i}, \hat{\rho}_i(k)) = \max\left(\eta_{\min,i}, \eta_{\min,i} e^{\left(\beta_i'\left(\frac{\hat{\rho}_i(k)}{\gamma_i'\rho_{\mathrm{crit},i}}-1\right)\right)}\right) \tag{5.43}$$

$$F_\mathrm{update}(\eta_i(k), \hat{\rho}_i(k)) = (1+\alpha_i') \cdot \max\left(\eta_i(k), \eta_i(k) e^{\left(\beta_i'\left(\frac{\hat{\rho}_i(k)}{\gamma_i'\rho_{\mathrm{crit},i}}-1\right)\right)}\right) \tag{5.44}$$

with $\rho_{\mathrm{crit},i}$ the critical density of cell i, $\alpha_i', \beta_i' > 0$ constants reflecting the importance of the traffic dynamics in cell i, and $0 < \gamma_i' \leqslant 1$ a threshold parameter for the critical density.

Note that an MDP is usually described in a discrete way, so the interest levels may be discretized into N_i levels. If so, the value of α_i' should be selected to be large enough such that the discretized value of the interest level will change once $\eta_i(k)$ is updated. Moreover, the number N_i should be selected as $N_\mathrm{i} > \alpha N_\mathrm{c}$ with $\alpha \approx 1$. Otherwise, the interest level on each cell will reach its maximum so quickly that there is no incentive for UAVs to visit different cells.

Reward function

As explained in Section 5.3.1, the UAVs are encouraged to visit the cells with high interest level, so the interest level is in fact the reward:

$$r_n(k) = \eta_i(k), \text{ if } p_n(k) = y_i. \tag{5.45}$$

5.3.3 Solution methods

The UAV path planning problem is formulated as a standard deterministic MDP problem, which can be solved by standard MDP solution methods, e.g., Q-iteration [9]. Generally speaking, the goal of the Q-iteration is to find an optimal Q-function, and the optimal control policy is derived by selecting at each state an action with the largest value of the optimal Q-function. More information about solution methods for MDP can be found in [12, 129].

The main problem when using Q-iteration for the given setting is that the Q-function is usually represented in a tabular form with one entry for each state-action pair (x, u). Therefore, when dealing with very large discrete or continuous state and action spaces, it is obviously impossible to store the value of the Q-function for every (x, u). Moreover, the computational cost when applied to a deterministic MDP with a finite number of states and actions is $|\mathscr{X}||\mathscr{U}|(2+|\mathscr{U}|)$ per iteration [26], where $\mathscr{X}$ denotes the state space, $\mathscr{U}$ denotes the action space, and $|\cdot|$ denotes the cardinality of the argument set. In the UAVs path planning problem, the number of states is

$$|\mathscr{X}| = N_{\mathrm{i}}^{N_{\mathrm{c}}} \cdot N_{\mathrm{c}}^{N_{\mathrm{u}}} \ , \tag{5.46}$$

with N_{i} the number of interest levels after discretization, N_{c} the number of cells, and N_{u} the number of UAVs. The number of actions is

$$|\mathscr{U}| = N_{\mathrm{a}}^{N_{\mathrm{u}}} \tag{5.47}$$

with N_{a} the number of possible actions at each state. Therefore, it is not only memory intensive, but also computationally expensive to directly use Q-iteration to solve the UAV path planning problem in a large-scale network.

In order to solve this problem in practice, some alternative solution methods are introduced, namely fitted Q-iteration, model predictive control, and parameterized control.

Fitted Q-iteration

One way to deal with very large discrete or with continuous state and action spaces is to approximate the Q-functions [116]. The fitted Q-iteration method [53] is an offline reinforcement learning algorithm that iteratively yields an approximation of the Q-function. The basic idea of this algorithm is to compute the Q-function by regressively approximating it based on a set $\mathscr{T}$ of training samples that consists of four-tuples (x, u, r, x') with x the current state, u the action, r the reward, and x' the next state. The algorithm is presented in Algorithm 5.1. It has two main steps for each training iteration t: (1) building the training set $\mathscr{S}_t$ and (2) training the Q-function Q_t. The training set $\mathscr{S}_t$ contains input-output pairs, where the input is the state-action pair (x, u), and the output is determined by using the standard Q-iteration rule based on the previous Q-function Q_{t-1}. The Q-function Q_t is then derived based on the

training set $\mathscr{S}_t$ by using a regression algorithm R, which is a supervised learning method, e.g., the extremely randomized trees approach [53], or an artificial neural network approach [120].

Algorithm 5.1 Fitted Q-iteration algorithm

Input: set of train samples $\mathscr{T}$, regression algorithm R, discount parameter γ, maximum iteration $T_{\max}$

initialize Q-function: $Q_0 \leftarrow 0$
$t \leftarrow 1$
repeat
  $\mathscr{S}_t \leftarrow \emptyset$
  for every $(x, u, r, x') \in \mathscr{T}$ **do**
    $o \leftarrow r + \gamma \max_{u'} Q_{t-1}(x', u')$
    $\mathscr{S}_t \leftarrow \mathscr{S}_t \cup \{((x, u), o)\}$
  end for
  train Q-function $Q_t = R(\mathscr{S}_t)$
  $t \leftarrow t + 1$
until $t = T_{\max}$

Output: Q-function $Q_{T_{\max}}$

The reasons to choose the fitted Q-iteration are:

- The method allows to use any approximator to fit the Q-function [53];
- The control policy can be successfully learned from relatively few training samples, so the method is data efficient.

Model predictive control

The objective of an MDP problem is in fact to solve an optimization problem that seeks to maximize the cumulative aggregation of rewards over a given time horizon. This idea is in the spirit of receding horizon techniques, which are associated with model predictive control (MPC). Therefore, a model-based MDP problem can be also solved by using MPC. A detailed explanation of MPC is out of the scope of this chapter, but interested readers are referred to [27, 95, 118].

For the specific problem in this chapter, the MPC process is described by:

1. **Prediction**: The prediction is made by repeatedly applying (5.41), (5.42), and (5.45) during the prediction period $[kT, (k+H_\mathrm{p})T]$, with H_p the prediction horizon. The inputs for the model-based prediction are the current position $p_n(k)$ for UAV n, the current interest level $\eta_i(k)$ on cell i, and the control vector $\boldsymbol{\theta}_n(k) = [\theta_n(k|k)\ \theta_n(k+1|k)\ \ldots\ \theta_n(k+H_\mathrm{c}-1|k)]^T$, where $\theta_n(\cdot|k)$ denotes the control action for the corresponding control step based on information available at time step k, and $H_\mathrm{c}(\leqslant H_\mathrm{p})$ denotes the control horizon. If H_c is smaller than H_p, the control action follows the constraint $\theta_n(k+k'|k) = \theta_n(k+H_\mathrm{c}-1|k)$ for $k' = H_\mathrm{c}, H_\mathrm{c}+1, \ldots, H_\mathrm{p}-1$. Based on these inputs, the

future evolution of the positions, interest levels and the rewards are predicted:

$$\hat{\boldsymbol{p}}_n(k) = [\hat{p}_n(k+1|k)\hat{p}_n(k+2|k)\dots\hat{p}_n(k+H_\mathrm{p}|k)] \tag{5.48}$$

$$\hat{\boldsymbol{\eta}}_i(k) = [\hat{\eta}_i(k+1|k)\hat{\eta}_i(k+2|k)\dots\hat{\eta}_i(k+H_\mathrm{p}|k)] \tag{5.49}$$

$$\hat{\boldsymbol{r}}_n(k) = [\hat{r}_n(k+1|k)\hat{r}_n(k+2|k)\dots\hat{r}_n(k+H_\mathrm{p}|k)] \tag{5.50}$$

2. **Optimization**: The goal is to find the optimal action vector $\boldsymbol{\theta}_n^*(k)$ to maximize the total rewards of all the UAVs during the prediction period $[kT,(k+H_\mathrm{p})T]$. Therefore, the objective function is formulated as:

$$J(k) = \sum_{k'=k+1}^{k+H_\mathrm{p}} \sum_{n=1}^{N_\mathrm{u}} r_n(k') \ . \tag{5.51}$$

The following optimization problem is then solved:

$$\begin{aligned} \max_{\boldsymbol{\theta}_n(k)} \ & J_\mathrm{MPC}(k) \\ \text{subject to} \ & \Phi_\mathrm{MPC}(\boldsymbol{\theta}_n(k)) = 0, \\ & \Psi_\mathrm{MPC}(\boldsymbol{\theta}_n(k)) \leqslant 0 \end{aligned} \tag{5.52}$$

where $\Phi_\mathrm{MPC}(\cdot)$ captures all the equality constraints, and $\Psi_\mathrm{MPC}(\cdot)$ captures all the inequality constraints. In general, the problem (5.52) is a real-valued nonconvex optimization problem, which can be solved by e.g. multi-start sequential quadratic programming [19], simulated annealing [83], genetic algorithms [5], ant colony optimization [46], and so on.

3. **Control action**: Only the first sample $\theta_n^*(k|k)$ of the optimal control action $\boldsymbol{\theta}_n^*(k)$ is applied to the process.

Then, the procedure from prediction to control action is repeated at time step $k+1$ with the prediction horizon shifted one time step ahead, and so on.

Parameterized control

The parameterized control proposed in this chapter is a heuristic method, where actions are determined at each time step by control rules that contain parameters. The goal is to optimize the parameters in the rules such that the rewards received by UAVs during a certain period are maximized. The advantage is that only a few parameters need to be optimized, so the approach is computationally efficient.

Intuitively, at each time step a UAV should strive for the cell i^* with the highest interest level $\eta_{i^*}(k)$ in the network in order to maximize the total reward. However, this may in fact reduce the total reward, e.g., if the distance between cell i^* and the current cell i is too large, or if the difference between the interest levels on cell i^* and cell i is too small. Therefore, a UAV should move not only with a global objective, but also with a local objective. A local objective for a UAV n at cell i could be characterized by a neighborhood set $\mathcal{N}_i$, and a reachable set $\mathcal{H}_{i,\theta}$. A neighborhood set $\mathcal{N}_i$ includes the cells that are adjacent to cell i. Moreover, if a UAV n moves *twice*[5] from cell i to cell j by taking the same action θ, the neighborhood

[5]The size of a reachable set can be easily increased by making the UAV move three or more times.

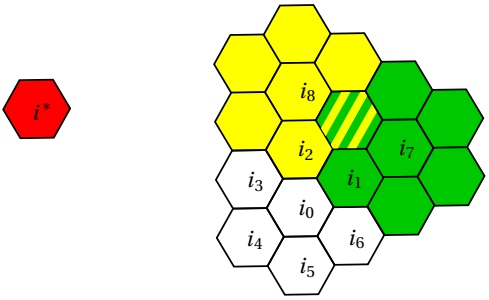


Figure 5.4: Illustration of neighborhood set and reachable set: i_0 is the current cell, $\mathcal{N}_{i_0} = \{i_1, i_2, i_3, i_4, i_5, i_6\}$ is the neighborhood set of i_0; the reachable set $\mathcal{H}_{i_0,\frac{\pi}{6}}$ includes all the green cells, and the reachable set $\mathcal{H}_{i_0,\frac{\pi}{2}}$ includes all the yellow cells (the overlapping cell of the sets $\mathcal{H}_{i_0,\frac{\pi}{6}}$ and $\mathcal{H}_{i_0,\frac{\pi}{2}}$ is indicated by both colors).

set $\mathcal{N}_j$ of cell j plus cell j itself is called the reachable set $\mathcal{H}_{i,\theta}$ (see Figure 5.4 for an example). The purpose of defining the reachable set is to let UAVs compare more cells than their immediate neighborhood at each step. From a local objective point-of-view, a UAV at cell i should select the action $\theta \in \Theta$ that results in the highest interest level in the corresponding reachable sets $\mathcal{H}_{i,\theta}$.

Supposing that the current position $p_n(k)$ of UAV n is $p_n(k) = y_i$, some global and local rules are proposed as follows (many other variations are possible):

- **Global rule**: with a probability $P(k) = \delta_P \cdot \frac{L}{\|p_n(k) - y_{i^*}\|} \cdot \frac{\eta_{i^*}(k) - \eta_i(k)}{\eta_{i^*}(k)}$, the next position is $p_n(k+1) = y_j$, for $j = \arg\min_{j' \in \mathcal{N}_i} \|\mathrm{h}(\overrightarrow{y_i y_{j'}}) - \mathrm{h}(\overrightarrow{y_i y_{i^*}})\|$;
- **Local rules**: with a probability $1 - P(k)$,
  1. If $\frac{\eta_j(k)}{\sum_{j' \in \mathcal{N}_i} \eta_{j'}(k)} \geqslant \delta_1$, for $j \in \mathcal{N}_i$, the next position is $p_n(k+1) = y_j$
  2. Else if $\frac{\sum_{j \in \mathcal{H}_{i,\theta}} (\eta_j(k))^{\delta_3}}{\sum_{\theta' \in \Theta} \sum_{j' \in \mathcal{H}_{i,\theta'}} (\eta_{j'}(k))^{\delta_3}} \geqslant \delta_2$, for $\theta \in \Theta$, the next position is $p_n(k+1) = f_{\mathrm{p}}(y_i, \theta)$, with f_{p} given by (5.41);
  3. Else, the next position is $p_n(k+1) = y_j$, for $j = \arg\max_{j' \in \mathcal{N}_i} \eta_{j'}(k)$;

with $\overrightarrow{y_i y_j}$ the vector from the point y_i to the point y_j, the operator $\mathrm{h}(\cdot)$ the heading of a vector, and $\delta_P, \delta_1, \delta_2 > 0$, and $\delta_3 > 1$ the parameters to be optimized.

The parameters δ_P and $\delta_1, \delta_2, \delta_3$ can be optimized online and offline. The online optimization can be the same as the aforementioned MPC method, through parameterization of the MPC inputs, while the offline optimization could consider a set of representative scenarios with different traffic densities of the freeway network, and the different configurations of the UAVs.

5.4 Case study

In this section, we provide a simulation example of the proposed UAV path planning strategies. The case study involves a part of the Singapore expressway network, as shown in Fig-

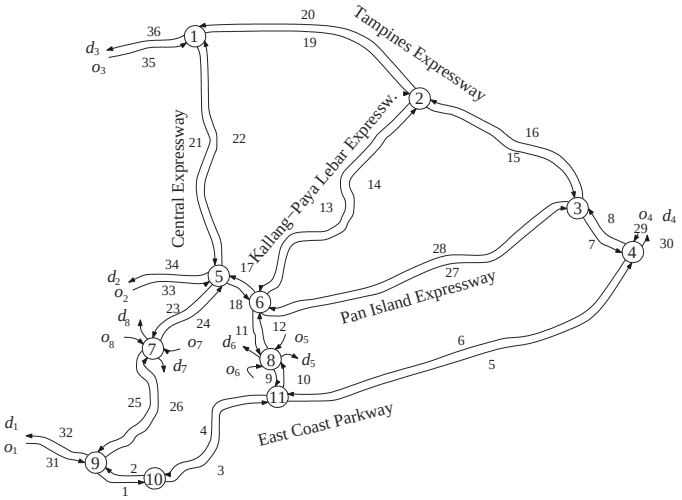


Figure 5.5: Central and eastern part of the Singapore Expressway Network.

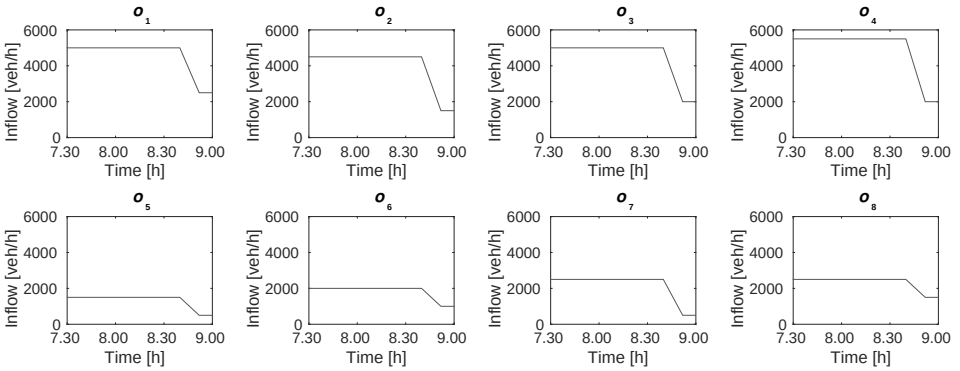


Figure 5.6: Traffic inflow of the origins during 7.30 *am to* 9.00 *am for the case study.*

ure 5.5. This area contains 8 origins, 8 destinations, and 18 highway roads, which are modeled as 36 one-way physical links in this case study. We use a dynamic traffic model to represent the Singapore expressway network. In particular, in order to obtain fast computations, the METANET model [98] is chosen as the simulation model. We set up a rush hour traffic scenario for this simulated freeway network, and let the UAVs monitor it. Then, the performance of the UAVs is evaluated for the two monitoring settings.

5.4.1 Simulation setting

Traffic scenario

For the case study, we consider the traffic conditions during a rush hour period between 7.30 am and 9.00 am, with the traffic inflow of each origin during this period shown in Figure 5.6. Moreover, in order to evaluate the monitoring performance of the UAVs in severe traffic situations, traffic congestion is created at the destinations d_5 and d_6 between 7.30 am to 8.00 am,

and at the destinations d_7 and d_8 between 7.30 am to 8.00 am respectively, by adding a large downstream density in the network with a value of 50 veh/km/lane as boundary conditions.

Parameters of the traffic simulation model

The METANET model (see Appendix B) is used to simulate the traffic flow evolution in this case study. The model parameters are set as [85]: length of each simulation step T = 10 s, length of each segment L_m = 500 m, free flow speed $v_{\text{free},m}$ = 90 km/h, critical density $\rho_{\text{crit},m}$ = 27 veh/km/lane, maximum density $\rho_{\text{max},m}$ = 110 veh/km/lane, link capacity C_m = 1500 veh/h/lane, and model parameters η = 60 km^2/h, κ = 40 veh/km/lane, and τ = 18 s.

UAV settings

Each UAV is modeled as a moving mass point in this case study, which means that we only consider its speed and energy consumption. We set the low speed[6] at v_{low} = 100 km/h, and the high speed at v_{high} = 300 km/h. The energy-speed function f_{e} in (5.9) is modeled by a second-order polynomial:

$$f_{\text{e}}(v) = av^2 + bv + c \ , \tag{5.53}$$

with $a = 0.5 \cdot 10^{-3}$, $b = 2 \cdot 10^{-3}$, and $c = 10^{-3}$.

Monitoring Setting I

Three UAVs are used to monitor the network in Setting I, and the length of each monitoring period is T_{m} = 0.5 h. The UAVs are put at three different depots at the beginning of the simulation. UAV 1 is put at depot 1, which is connected to origin o_1, UAV 2 is put at depot 2, which is connected to origin o_3, and UAV 3 is put at depot 3, which is connected to origin o_4.

For the first monitoring period, we select the links in the central business area (links 1, 2, 3, 4, 9, 10, 11, 12, 17, 18, 23, 24, 25, and 26), and the links that enter this area (links 5, 13, 21, 28, 31, 33, and 35) as the links that must be monitored, and these links are mapped into $\mathcal{V}_{\text{moni,high}}$. The remaining links are mapped into $\mathcal{V}_{\text{moni,mod}}$. The initial value of the attraction level for links in $\mathcal{V}_{\text{moni,high}}$ is set equal to Φ_{high} = 0.5, and the initial value of the attraction level for links in $\mathcal{V}_{\text{moni,mod}}$ is set equal to Φ_{low} = 0.25 (see Figure 5.7(a)). For the rate of increase of the attraction level, the parameter α_i in (5.5) for the links in the central business area is set equal to 2, and for the remaining links it is set equal to 1. The parameter β_i is set equal to 2, and the parameter γ_i is set equal to 0.95.

For the objective function (5.8), the weighting parameters are: λ_1 = 0.001, λ_2 = 0.7, λ_3 = 0.1, λ_4 = 0.2, and λ_5 = 0.01.

The MILP problem was solved via the Tomlab toolbox for Matlab using CPLEX.

Monitoring Setting II

In this setting, the number of the UAVs, the location of each depot, and the length of the monitoring period are chosen the same as the ones in Setting I. In Setting II, the UAVs only fly with a constant speed v_{high} = 300 km/h from one cell to another, and pause at the center of each cell to monitor. The radius of each hexagonal cell is R = 500 m.

[6] Usually, the speed of civilian UAVs is between 30 km/h and 400 km/h, and the speed of military UAVs is between 200 km/h and 800 km/h.

Since the METANET model divides a link into multiple segments, the traffic density $\rho_i(k)$ of each non-empty cell i equals the average traffic density of the segments that are covered by cell i. If a segment is partially covered by a cell, we only consider the traffic situation on the covered part. We define a parameter $0 < p_{i,m,j} \leqslant 1$ to indicate the percentage of the part that segment j of link m is covered by cell i. In this way, the traffic density $\rho_i(k)$ is calculated as:

$$\rho_i(k) = \frac{\sum\limits_{(m,j)\in\mathscr{S}_i} p_{i,m,j}\lambda_m\rho_{m,j}(k)L_m}{\sum\limits_{(m,j)\in\mathscr{S}_i} p_{i,m,j}\lambda_m L_m}\ , \tag{5.54}$$

with $\rho_{m,j}(k)$ the traffic density of segment j in link m at time step k, $\mathscr{S}_i$ the set of pairs of indices (m, j) of the links and segments that are covered by cell i, L_m the length of segments in link m, and λ_m the number of lanes of link m.

The minimum interest level $\eta_{\min,i}$ in (5.43) is set equal to 0, the parameters α_i', β_i', and γ_i' in (5.44) are set equal to 1, 1.5, and 1 respectively. The number of the discretized interest levels is $N_\mathrm{i} = 50$.

For the solution methods, the regression algorithm used in the fitted Q-iteration is the regression tree package created by [64], and the optimization algorithm used in MPC and the parameterized control is the genetic algorithm implemented via the ga function of the Matlab Global Optimization Toolbox.

5.4.2 Simulation results

Setting I

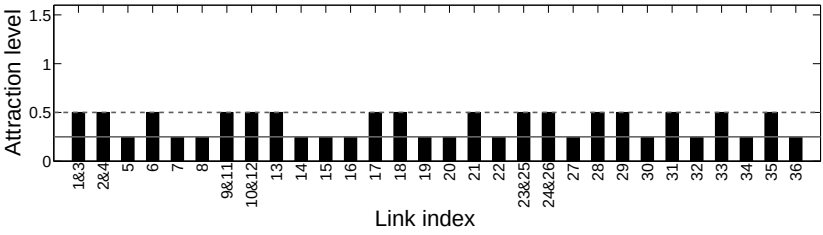
In order to reduce the computation cost of the optimization problem, the number of the optimization variables can be decreased by merging some links if these links are concatenated. For the Singapore expressway network, we merge links 1 and 3, links 2 and 4, links 9 and 11, links 10 and 12, links 23 and 25, and links 24 and 26 respectively. In this way, the number of the physical links for the case study is decreased to 30.

The results of the attraction levels on each link at the beginning of each monitoring period are shown in Figure 5.7. If the value of the attraction level is above Φ_{high} (indicated by the red dashed line), the corresponding link must be monitored in the upcoming period; if the value of the attraction level is between Φ_{high} and Φ_{low} (indicated by the green solid line), the corresponding link may be monitored in the upcoming period; else, the corresponding link may only be used by UAVs for traversing in the upcoming period.

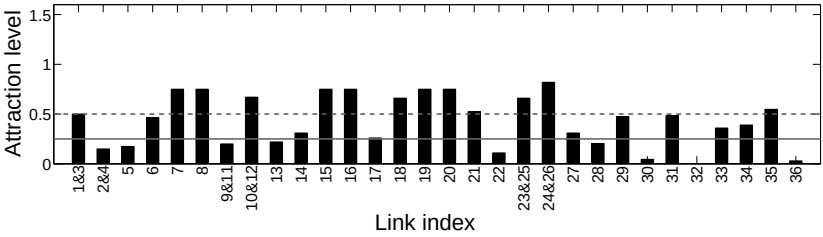
The results of the proposed path planning algorithm for UAVs are shown in Figure 5.8. Each subfigure shows the trajectories of the UAVs when monitoring the network, and each table shows the order of the links that are visited by the UAVs. We consider the case that the UAVs can return to any depot when finishing the surveillance; so, at the beginning of a new monitoring period, each UAV starts from the depot at which it ended in the previous period.

Setting II

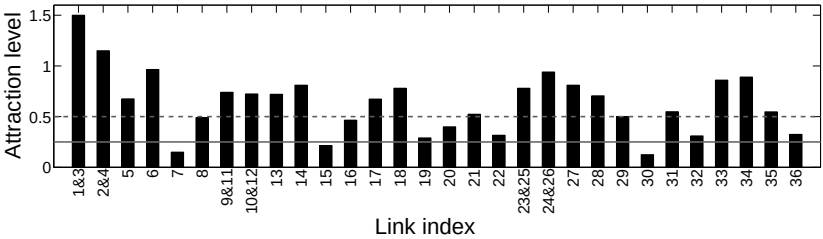
The network is decomposed into multiple hexagonal cells as shown in Figure 5.9. The dark cells indicate the empty cells that do not cover any freeway link, so only the white cells are required to be monitored.



(a) Attraction levels at 7.30 am

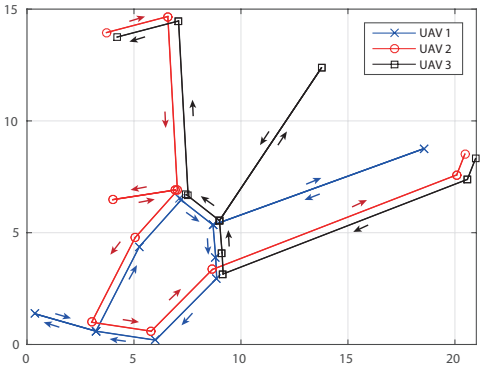


(b) Attraction levels at 8.00 am



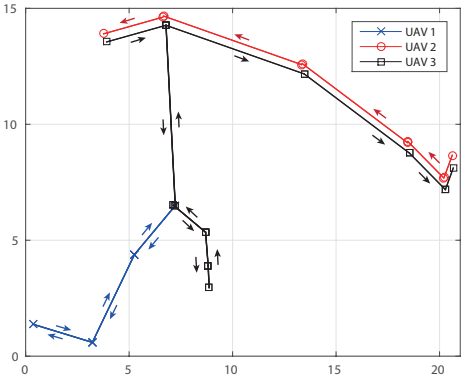
(c) Attraction levels at 8.30 am

Figure 5.7: Simulation result: attraction levels at the beginning of different monitoring periods. The red dashed line indicates the threshold Φ_{high}, and the green solid line indicates the threshold Φ_{low}.



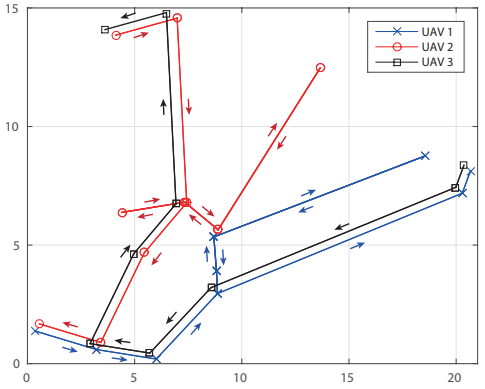
UAV 1	31 → 26 → 24 → 18 → 27 → 28 → 11 → 9 → 4 → 2 → 32
UAV 2	35 → 21 → 34 → 33 → 23 → 25 → 1 → 3 → 5 → 30
UAV 3	29 → 6 → 10 → 12 → 14 → 13 → 17 → 22 → 36

(a) Simulation results: path planning for UAVs between 7.30 am and 8.00 am



UAV 1	31 → 26 → 24 → 23 → 25 → 32
UAV 2	29 → 8 → 16 → 20 → 36
UAV 3	35 → 21 → 18 → 11 → 9 → 10 → 12 → 17 → 22 → 19 → 15 → 7 → 30

(b) Path planning results for UAVs between 8.00 am and 8.30 am



UAV 1	31 → 1 → 3 → 10 → 12 → 27 → 28 → 11 → 9 → 5 → 30
UAV 2	35 → 21 → 18 → 14 → 13 → 17 → 34 → 33 → 23 → 25 → 32
UAV 3	29 → 6 → 4 → 2 → 26 → 24 → 22 → 36

(c) Path planning results for UAVs between 8.30 am and 9.00 am

Figure 5.8: Simulation results of setting 1 by using MILP.

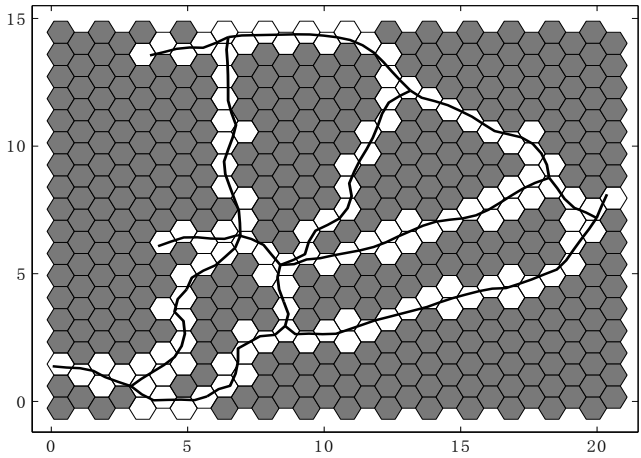
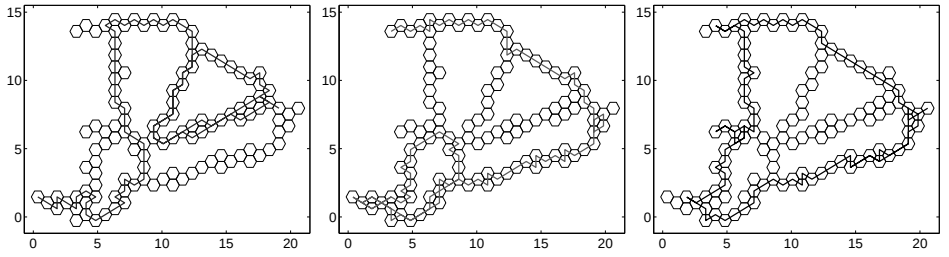


Figure 5.9: Hexagonal decomposition of the Singapore expressway network.

For the sake of compactness, we only show the results of the first monitoring period; the results of the other periods are similar. The trajectories of the UAVs obtained using the three different approaches are presented in Figures 5.10 – 5.12, and the computation time and the monitoring performance are listed in Table 5.3. Comparing the computation time, fitted Q-iteration is the fastest approach, because the Q-function was yielded by offline training before simulation; on the other hand, MPC is much slower than the other two approaches, because the prediction of the future states and the future control actions of the UAVs result in extra computation time. Comparing the monitoring performance, we introduce two criteria for evaluation. The first criterion is the covering percentage, which is computed by dividing the number of cells covered by the UAVs during the monitoring period by the total number of non-empty cells. As shown in Table 5.3, fitted Q-iteration results in the best covering performance, a full coverage, while the other two approaches only have a coverage rate of 98.35%. The second criterion is the monitoring quality, i.e., the degree to which the traffic situations are monitored by each UAV. Note that this quality is not the same as the reward received by each UAV. The reason is that the reward computed by (5.42)-(5.45) is also related to when the cell is monitored, and the importance of the cell. In order to only compare the traffic situations monitored, we define the monitoring quality as follows: at every step k, if the traffic density $\hat{\rho}_i(k)$ of cell i monitored by UAV n is below the critical density $\rho_{\text{crit},i}$, the quality is 1; otherwise, the quality is computed by:

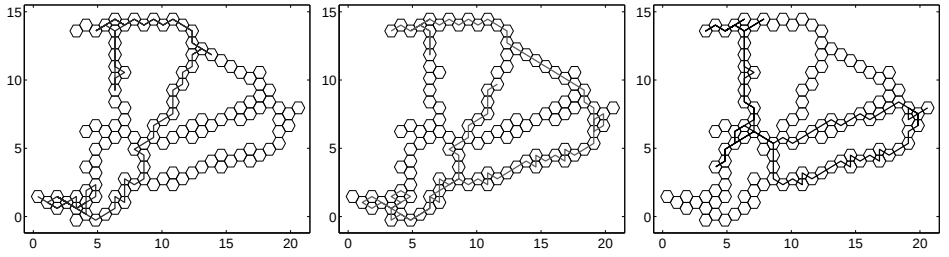
$$\mu_n(k) = \max\left(1, \frac{\hat{\rho}_i(k)}{\rho_{\text{crit},i}}\right), \text{ if } p_n(k) = y_i. \tag{5.55}$$

Comparing the three different approaches, the performance obtained by using MPC is a little better, because the sum of the monitoring qualities is higher than the ones of the other two approaches. Moreover, by using MPC, the monitoring quality of each UAV is almost the same, while for the other two approaches, UAV 1 is better than UAV 2 and UAV 3. To summarize, for the given case study, the fitted Q-iteration approach has a better coverage performance, but MPC performs better on the monitoring quality, but in general, the performance results of these three approaches are similar with each other.



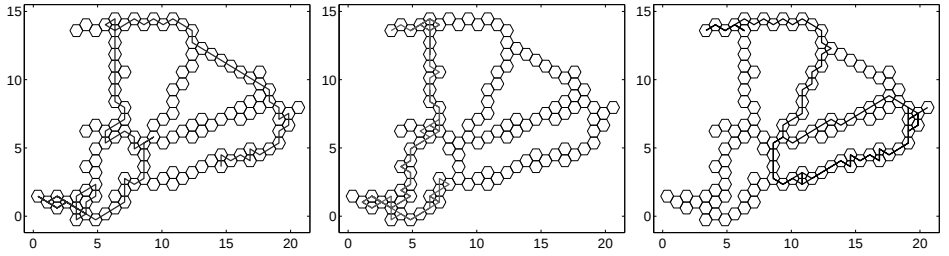
(a) Trajectory of UAV 1 (b) Trajectory of UAV 2 (c) Trajectory of UAV 3

Figure 5.10: The trajectories of the UAVs yielded by the fitted Q-iteration in the first monitoring period.



(a) Trajectory of UAV 1 (b) Trajectory of UAV 2 (c) Trajectory of UAV 3

Figure 5.11: The trajectories of the UAVs yielded by the MPC in the first monitoring period.



(a) Trajectory of UAV 1 (b) Trajectory of UAV 2 (c) Trajectory of UAV 3

Figure 5.12: The trajectories of the UAVs yielded by the parameterized control in the first monitoring period.

Table 5.3: Monitoring performance of using three different approaches in the first monitoring period

Solution methods	Computation time [s]	Covering percentage	Average monitoring quality		
			UAV 1	UAV 2	UAV 3
Fitted Q-iteration	1736.3	100%	1.25	1.12	1.07
MPC	8246.3	98.35%	1.15	1.18	1.16
Parameterized control	2062.7	98.35%	1.26	1.03	1.10

Note that we do not consider that UAVs must return to a depot before the end of the monitoring period as in Setting I, but just let the UAVs stop at the end of each monitoring period, and afterwards they directly fly back to their own depot for recharging or refuel. In order to prevent that UAVs are too far away from the depots at the end of the monitoring period, one can add an endpoint penalty to the reward, based on the distances between the depots and the location of each UAV when finishing the surveillance. The longer the distance is, the higher the penalty is.

5.5 Conclusions

This chapter has addressed the path planning problem of unmanned aerial vehicles (UAVs) for monitoring freeway networks. We have considered two distinct monitoring settings: in the first setting, the UAVs have two flying modes — monitoring and traversing, while in the second setting, the UAVs only have one flying mode, so they can only monitor when hovering in the air. For the first setting, the monitoring problem has been formulated as a periodical multiple rural postman problem, and solved using Mixed-Integer Linear Programming (MILP). For the second setting, the problem has been stated as a Markov Decision Process (MDP), and three solution methods, namely the fitted Q-iteration, the Model Predictive Control, and the parameterized control, have been proposed. A case study involving a part of the Singapore expressway network has shown that our proposed algorithms can find near full coverage paths for the network monitoring problem. A common idea used in both settings is to introduce an auxiliary variable to indicate the need of a link/cell to be monitored, called the attraction level and the interest level respectively, and they both work as a medium on each link/cell providing dynamic traffic information for UAVs.

Chapter 6

Conclusions and Future Research

This thesis has presented several approaches for traffic management and control in freeway networks, with the goal of improving the network performance, as well as decreasing the computational burden. In this chapter, the contributions of the thesis are summarized in Section 6.1, the main conclusions are given in Section 6.2, and some open issues that can be investigated in the future are highlighted in Section 6.3.

6.1 Contributions

The main contributions of this thesis are summarized as follows:

- We have developed an ant-based algorithm to solve the dynamic traffic routing problem in freeway networks, including two main concepts: stench pheromone, which is used to expel ants from crowded arcs, and colored ants, which are used for a network with multiple destinations. Moreover, we have also proposed how to use artificial ants to calculate the dynamic link cost when traffic conditions are changing on that link.
- We have defined a unified problem formulation for co-design of network topology and traffic control measures in a model-based optimization framework, where the network topology design and the traffic control measures are jointly optimized.
- We have considered the monitoring problem for Unmanned Aerial Vehicles (UAVs) in two different settings: the first setting is formulated as a periodical multiple rural postman problem, and the second setting is formulated as a Markov Decision Process (MDP). We have introduced an attraction/interested level in both settings to indicate the importance of each link in the network.

6.2 Conclusions

For a traffic control approach, improving the network performance and reducing the computational complexity are in general two conflicting objectives: a better performance usually comes at the cost of a higher computational burden. The complexity of a traffic control approach could increase exponentially with the number of nodes and links in the network. Therefore, when dealing with a large-scale network, it is possible that the problem cannot be solved in practice due to the limitations of the computational capabilities of the available

hardware, although theoretically the control approaches might be able to find the optimal solutions, such as the fully-dynamic approach for the Ant Colony Routing (ACR) algorithm proposed in Chapter 3, or the standard Markov Decision Process (MDP) solution methods for the UAV path planning problem proposed in Chapter 5. Therefore, a general conclusion of this thesis is that for an appropriate traffic control approach, a well-balanced trade-off between the network performance and the computational efficiency is more important than only considering the performance or the computational efficiency independently.

A few guidelines for choosing an appropriate traffic control approach w.r.t. achieving a well-balanced trade-off between the performance and the computational efficiency are provided next:

- **Reducing the size of the network**:
  In order to reduce the computational burden of a control approach, the most straightforward way is to apply that approach to a reduced network. For a large-scale network, it is possible that some of the links are used by less drivers, so compared to the other links, these links should not necessarily be controlled. In this way, we can temporarily remove them from the network, and only apply the control approach for the remaining part. Note that this method can be applied multiple times for the same network, e.g., for each origin-destination pair, or at each control step.

  This thesis gives some examples that show how to appropriately reduce the size of the network. In Chapter 3, we use a linear programming method to find the links with the highest flows to determine the pruned network; in Chapter 5, we define an auxiliary variable called attraction level, based on the traffic density of each link, to determine whether the corresponding link is chosen to be monitored in the next period. Generally speaking, the goal is to remove the less important links, and the importance of each link is usually associated with the traffic conditions on that link, or it can be manually defined by the traffic authorities.

- **Solving the problem in a hierarchical way**:
  When dealing with a multi-objective problem, one can divide it into multiple levels, and then solve them in a hierarchical way, which means that the different levels of the problem are solved step by step.

  Chapter 4 divides the co-design problem for a traffic network into two levels: topology design and control measures design, and compares four different solution frameworks. In the case study considered in this thesis, the separate optimization framework yields the worst result, with the total cost being about 28% higher than the other three frameworks. This is because the separate optimization framework solves the two levels of the problem independently, not considering any relationship between them. The other three frameworks yield results that are similar to each other. The iterative solution framework solves the two levels in an alternating way; the bi-level optimization framework embeds a lower level into an upper level optimization problem, and focuses on solving the upper level of the problem; the joint optimization framework solves the two levels as a whole, which can be considered as a special case of the hierarchical solution framework. Although in the case study the results of these three solution frameworks are similar, it should be emphasized that the iterative solution framework may never converge to the optimal solution itself. Therefore, solving a problem in a hierarchical way is indeed helpful for improving the network performance and the computational

efficiency, but there is no general rule of which framework is the best. One should choose the most appropriate framework on a case-by-case basis.

- **Reducing the number of control/optimization variables**:
  In traffic control, a control signal is applied to the network at every control step. The computational burden of a control approach can be decreased by reducing the number of the control variables. One way to do that is to make the control step interval longer so that there are fewer control steps during the entire control period. Moreover, by using model predictive control, one can manipulate the trade-off between the performance and the computation time by changing the length of the prediction horizon or the control horizon. However, if the entire control period is too long, these methods may still have a high computational burden. Therefore, this thesis considers another method — parameterized control, which optimizes the parameters of the control laws instead of the control signals themselves. Usually, the parameters are time-independent, so their number is much less than the number of the control signals. Hence, the computation time is reduced.

  In Chapter 2, we aim at finding the optimal parameters of the stench pheromone function for the ACO-SP algorithm. In the case study, ACO-SP is four times faster than the optimal control method, while obtaining almost the same total travel time as the latter one. In Chapter 4, we use parameterized control to bring the network topology design and the control measures design to the same time scale. In this way, the optimization variables only involve the parameters of the control measures, instead of the time-varying control signals.

6.3 Recommendations for Future Research

In this section, we give some recommendations for possible future research directions:

- **Alternatives for ACO-SP**:
  In order to disperse ants in an ant graph, one can consider how to construct better stench pheromone functions in ACO-SP, or even develop a totally different mechanism. For example, instead of using the stench pheromone, one can let the pheromone evaporation rate depend on the number of ants on each arc. If too many ants choose the same arc, the pheromone on that arc will then evaporate faster, resulting in a decline of the pheromone level, such that the probability that subsequent ants will choose that arc decreases accordingly, and vice versa. Alternatively, one can just adopt the collision mechanism that Lasius niger ants use to distribute themselves [49]. When an ant is going to choose an arc, it may collide with another ant; if a collision occurs, that ant is immediately pushed to another arc. The probability that a collision will happen is based on the number of ants on the arc that the ant is going to choose. The more ants are on that arc, the more possible it is that a collision will occur. Hence, the ants will be distributed over the ant graph.

- **Further extension of the co-design method**:
  When using the co-design method, one may only take a limited number of control measures into account during the design phase. However, it is possible that new control measures will be added in the network in the future. In this case, the robustness of

the co-design method should be taken into account, preventing the newly added control measures from having a negative impact on the network performance. Moreover, the proposed co-design method could be difficult to apply in case of a control measure that is hard to parameterize, or that has too many parameters. Therefore, one can also consider co-design methods using non-parameterized control, while the time scales of the network topology and control measures can still stay at the same level.

- **Implementations of UAVs path planning**:
  It is still an open question what is the better way to implement path planning for UAVs in real life. Currently, the use in practice of UAVs for traffic monitoring mainly focuses on the surveillance performance, not on path planning. Usually, the UAVs are manually assigned to fly over a pre-defined area, such as a straight line, or a circle. However, for improving performance, one can consider to let UAVs make their own choice of the roads or the areas to be monitored. For example, UAVs can decide based on both the short-term information, e.g., the traffic conditions just collected, and the long-term information, e.g., the historical data of the traffic network. Moreover, if multiple UAVs are simultaneously used, they might be able to share their own information with each other, and cooperate.

- **Driverless cars using ACO**:
  A future application for ACO in traffic control could be path planning for driverless cars, e.g., the Google car [65]. Compared to human-driven cars, driverless cars could be much more predictable and obedient, which makes them act more like ants. When traveling on the roads, each car might deposit a pheromone, which contains information such as travel time, travel speed, weather, and so on. Such a pheromone could be stored in some roadside facilities, and it could evaporate as time elapses. The roadside facilities are able to communicate with each other locally to share pheromone, such that information on any road can be sent upstream to the bifurcation points, where subsequent cars will make their own choice based on the pheromone level on each alternative road, just like the behavior of ants in the ACO algorithm. In this way, no central management system is needed. The driverless cars are more than just vehicles, but they also play roles of distributed sensors. The challenges in such an application are tremendous. For instance, it requires a robust and adaptive online extension of the proposed ACR algorithm, and fast communication and computation abilities for driverless cars. Some other problems could be how to guide the ACR cars in mixed networks where some links can collect, store and propagate pheromone, and others cannot, how to collect pheromones from regular cars that cannot deposit pheromone by themselves, and so on.

Appendix A

Convergence Proof of ACO-SP in a Two-Arc Graph

The research presented in this appendix is the first step towards a convergence proof of the Ant Colony Optimization with Stench Pheromone (ACO-SP) algorithm introduced in Section 2. It analytically proves the convergence property of ACO-SP in a graph with two arcs. This appendix is based on [36].

The convergence analysis of ACO algorithms has been introduced in [66, 67, 127]. Gutjahr [66, 67] presented a convergence proof for an ACO algorithm called graph-based ant system. The proof shows that the algorithm can generate an optimal solution at least once during the optimization. Stützle and Dorigo [127] prove that the $ACO_{gb,\tau_{min}}$ algorithm, which employs the global-best update rule, asymptotically converges to the optimal solution. However, because of the complexity and the diversity of ACO algorithms, there is no a general method to prove convergence for the entire class of ACO algorithms.

This appendix is structured as follows. Section A.1 states the convergence problem, as well as the assumptions for the proof. In particular, we introduce an auxiliary function that is used for the convergence proof. Next, the convergence of the auxiliary function is proved in Section A.2. Consequently, the convergence of the pheromone levels of ACO-SP in a two-arc graph is proved in Section A.3.

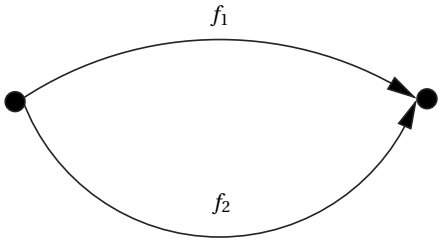


Figure A.1: A simple graph with two arcs.

A.1 Problem Statement

A.1.1 Notations and formulations

Definition A.1 *The ACO-SP algorithm is said to converge, if the pheromone levels on all arcs converge as the iteration step $t \to \infty$.*

As the first step, this appendix investigates a simple graph with two arcs as shown in Figure A.1. Without loss of generality, it is assumed that arc 1 is a better solution than arc 2, so the value of the fitness function f_1 of arc 1 is larger than the value of the fitness function f_2 of arc 2:

Assumption A.1 $f_1 > f_2$.

Before proceeding with the convergence proof, some important notations are first introduced. The pheromone levels in iteration t on arc 1 and 2 are denoted by $\tau_1(t)$ and $\tau_2(t)$, respectively. According to (2.12) and (2.14), the pheromone update equation is adapted as:

$$\begin{aligned}\tau_1(t+1) &= (1-\rho_{\text{evap}})\tau_1(t) + n_1(t)\cdot f_1 - \max(0, P(n_1(t)-N_1)) \\ \tau_2(t+1) &= (1-\rho_{\text{evap}})\tau_2(t) + n_2(t)\cdot f_2 - \max(0, P(n_2(t)-N_2))\end{aligned} \tag{A.1}$$

with $n_1(t)$ and $n_2(t)$ the numbers of ants that chose arc 1 and arc 2 in iteration t, and N_1 and N_2 the threshold numbers, respectively. Since there are only two arcs in the graph, it always holds that $n_1(t) + n_2(t) = N_{\text{ants,total}},\ t = 1, 2, \ldots$ According to the attraction mechanism of the ACO algorithm, the arc on which more pheromone has been accumulated has a higher probability to be chosen by ants. The probabilities $p_1(t)$ and $p_2(t)$ for selecting respectively arc 1 and arc 2 are computed based on (2.15):

$$\begin{aligned}p_1(t) &= \frac{\max(\tau_{\min}, \tau_1(t))}{\max(\tau_{\min}, \tau_1(t)) + \max(\tau_{\min}, \tau_2(t))} \\ p_2(t) &= \frac{\max(\tau_{\min}, \tau_2(t))}{\max(\tau_{\min}, \tau_1(t)) + \max(\tau_{\min}, \tau_2(t))}\end{aligned} \tag{A.2}$$

For illustration purposes, the value of α is set as $\alpha = 1$, but the proof is similar for $\alpha \geqslant 1$. For the parameters of the ACO-SP algorithm, the following assumptions are made:

Assumption A.2

1. $P \leqslant \min\left(\frac{N_{\text{ants,total}} \cdot f_1 - \rho_{\text{evap}}\tau_{\min}}{N_{\text{ants,total}} - N_1}, \frac{N_{\text{ants,total}} \cdot f_2 - \rho_{\text{evap}}\tau_{\min}}{N_{\text{ants,total}} - N_2}\right)$
2. $\tau_{\min} \leqslant \frac{f_2}{\rho_{\text{evap}}}$ *and* $\tau_{\min} \leqslant \tau_0$
3. $N_{\text{ants,total}}$ *is sufficiently large such that* $n_1(t), n_2(t) \geqslant 1,\ \forall t$
4. $N_{\text{ants,total}} \neq N_1$ *and* $N_{\text{ants,total}} \neq N_2$

Lemma A.1 *If Assumptions A.2.1 to A.2.3 are satisfied, it holds that* $\tau_1(t), \tau_2(t) \geqslant \tau_{\min}, \forall t$.

Proof: For $t = 0$, $\tau_1(t) = \tau_2(t) = \tau_0 \geqslant \tau_{\min}$ according to Assumption A.2.2. For $t = 1, 2, \ldots$ the proof will be done by introducing 3 cases:

Case A. If there is no stench pheromone, the pheromone update equations are given by:

$$\begin{aligned} \tau_1(t+1) &= (1-\rho_{\text{evap}})\tau_1(t) + n_1(t) \cdot f_1 \\ \tau_2(t+1) &= (1-\rho_{\text{evap}})\tau_2(t) + n_2(t) \cdot f_2 \end{aligned} \tag{A.3}$$

To prove that $\tau_1(t+1), \tau_2(t+1) \geqslant \tau_{\min}$ when $\tau_1(t), \tau_2(t) \geqslant \tau_{\min}$, it should hold that

$$\begin{aligned} (1-\rho_{\text{evap}})\tau_1(t) + n_1(t) \cdot f_1 &\geqslant \tau_{\min} \\ (1-\rho_{\text{evap}})\tau_2(t) + n_2(t) \cdot f_2 &\geqslant \tau_{\min} \end{aligned} \tag{A.4}$$

According to Assumption A.2.3, $n_1(t) \geqslant 1$ and $n_2(t) \geqslant 1$. Therefore, a sufficient condition for (A.4) to hold is

$$\begin{aligned} (1-\rho_{\text{evap}})\tau_{\min} + f_1 &\geqslant \tau_{\min} \\ (1-\rho_{\text{evap}})\tau_{\min} + f_2 &\geqslant \tau_{\min} \end{aligned} \tag{A.5}$$

To satisfy the inequalities above, it is required that $\tau_{\min} \leqslant \frac{f_1}{\rho_{\text{evap}}}$ and $\tau_{\min} \leqslant \frac{f_2}{\rho_{\text{evap}}}$. Because of Assumption A.1, we only need $\tau_{\min} \leqslant \frac{f_2}{\rho_{\text{evap}}}$ (Assumption A.2.2) as the sufficient condition.

Case B. If the stench pheromone is deposited on arc 1, i.e., $n_1(t) > N_1$, then according to (2.12) and (2.14), the pheromone update equation is given by:

$$\tau_1(t+1) = (1-\rho_{\text{evap}})\tau_1(t) + n_1(t) \cdot f_1 - P(n_1(t) - N_1) \tag{A.6}$$

To prove that $\tau_1(t+1) \geqslant \tau_{\min}$, when $\tau_1(t) \geqslant \tau_{\min}$, it should hold that

$$(1-\rho_{\text{evap}})\tau_1(t) + n_1(t) \cdot f_1 - P(n_1(t) - N_1) \geqslant \tau_{\min} \tag{A.7}$$

This inequality holds if

$$\begin{aligned} P &\leqslant \frac{(1-\rho_{\text{evap}})\tau_1(t) - \tau_{\min} + n_1(t) \cdot f_1}{n_1(t) - N_1} \\ &\leqslant \frac{(1-\rho_{\text{evap}})\tau_1(t) - \tau_{\min}}{n_1(t) - N_1} + \frac{n_1(t) \cdot f_1}{n_1(t) - N_1} \end{aligned} \tag{A.8}$$

Given a rational function defined by $y(x) = \frac{ax}{x-b}$, with $x > b$ and $a, b > 0$, it is easy to verify that $y(\cdot)$ is a monotonically decreasing function. Since $\tau_1(t) \geqslant \tau_{\min}$ and $n_1(t) \leqslant N_{\text{ants,total}}$, a sufficient condition for (A.8) to hold is

$$\begin{aligned} P &\leqslant \frac{(1-\rho_{\text{evap}})\tau_{\min} - \tau_{\min}}{N_{\text{ants,total}} - N_1} + \frac{N_{\text{ants,total}} \cdot f_1}{N_{\text{ants,total}} - N_1} \\ &\leqslant \frac{N_{\text{ants,total}} \cdot f_1 - \rho_{\text{evap}}\tau_{\min}}{N_{\text{ants,total}} - N_1} \end{aligned} \tag{A.9}$$

Case C. Similarly, if stench pheromone is deposited on arc 2, a sufficient condition for $\tau_2(t) \geqslant \tau_{\min}$ is that

$$P \leqslant \frac{N_{\text{ants,total}} \cdot f_2 - \rho_{\text{evap}}\tau_{\min}}{N_{\text{ants,total}} - N_2} \tag{A.10}$$

This holds by Assumption 2.1. □

With Lemma A.1, (A.2) can be further simplified as:

$$p_1(t) = \frac{\tau_1(t)}{\tau_1(t) + \tau_2(t)} \text{ and } p_2(t) = \frac{\tau_2(t)}{\tau_1(t) + \tau_2(t)} \tag{A.11}$$

The expected values of the numbers of ants that will select arc 1 and arc 2 in iteration t can be computed based on (A.11):

$$\begin{aligned} n_1(t) &= p_1(t) \cdot N_{\text{ants,total}} = \frac{\tau_1(t)}{\tau_1(t) + \tau_2(t)} \cdot N_{\text{ants,total}} \\ n_2(t) &= p_2(t) \cdot N_{\text{ants,total}} = \frac{\tau_2(t)}{\tau_1(t) + \tau_2(t)} \cdot N_{\text{ants,total}} \end{aligned} \tag{A.12}$$

A.1.2 Cases and Modes

According to the relationship between the total number $N_{\text{ants,total}}$ of ants and the threshold numbers of ants on arcs 1 and 2, N_1 and N_2, there are four different cases:

- Case 1: $N_{\text{ants,total}} < \min(N_1, N_2)$;
- Case 2: $N_1 < N_{\text{ants,total}} < N_2$;
- Case 3: $N_2 < N_{\text{ants,total}} < N_1$;
- Case 4: $N_{\text{ants,total}} > \max(N_1, N_2)$;

where $N_{\text{ants,total}}$ neither equals to N_1 nor N_2 due to Assumption 2.4. In each case, the process of updating pheromone is divided into four different modes, based on whether stench pheromone is deposited or not on the arcs:

- M1: No stench pheromone is deposited;
- M2: Stench pheromone is only deposited on arc 1;
- M3: Stench pheromone is only deposited on arc 2;

Case 1: $N_{\text{ants,total}} < \min(N_1, N_2)$	Case 2: $N_1 < N_{\text{ants,total}} < N_2$
M1 : $F(t) > 0$	M1 : $0 < F(t) \leqslant F_{\text{b,1}}$
M2 : NR	M2 : $F(t) > F_{\text{b,1}}$
M3 : NR	M3 : NR
M4 : NR	M4 : NR
Case 3: $N_2 < N_{\text{ants,total}} < N_1$	Case 4: $N_{\text{ants,total}} > \max(N_1, N_2)$
M1 : $F(t) \geqslant F_{\text{b,2}}$	M1 : $F_{\text{b,2}} \leqslant F(t) \leqslant F_{\text{b,1}}$
M2 : NR	M2 : $F(t) > \max(F_{\text{b,1}}, F_{\text{b,2}})$
M3 : $0 < F(t) < F_{\text{b,2}}$	M3 : $0 < F(t) < \min(F_{\text{b,1}}, F_{\text{b,2}})$
M4 : NR	M4 : $F_{\text{b,1}} < F(t) < F_{\text{b,2}}$

Table A.1: Defining four modes using $F(t)$. NR indicates "not reachable"

- M4: Stench pheromone is deposited on both arcs.

The four modes M1 to M4 can be redefined by introducing an auxiliary function $F(t) = \tau_1(t)/\tau_2(t)$. Since both $\tau_1(t)$ and $\tau_2(t)$ are positive, $F(t) > 0$ for all t. Taking M1 as an example, if there is no stench pheromone deposited on either of the arcs, the numbers $n_1(t)$ and $n_2(t)$ are not larger than the corresponding threshold number on each arc, i.e., $n_1(t) \leqslant N_1$ and $n_2(t) \leqslant N_2$. By using (A.12), we have:

$$\begin{aligned} \frac{\tau_1(t)}{\tau_1(t)+\tau_2(t)} N_{\text{ants,total}} &= \frac{F(t)}{F(t)+1} N_{\text{ants,total}} \leqslant N_1, \\ \frac{\tau_2(t)}{\tau_1(t)+\tau_2(t)} N_{\text{ants,total}} &= \frac{1}{F(t)+1} N_{\text{ants,total}} \leqslant N_2. \end{aligned} \tag{A.13}$$

Simplifying (A.13), it obtains:

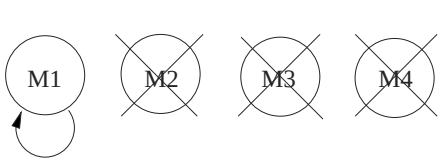
$$F(t) \geqslant \frac{N_1}{N_{\text{ants,total}} - N_1} \text{ and } F(t) \geqslant \frac{N_{\text{ants,total}} - N_2}{N_2} \tag{A.14}$$

For the sake of compactness, let $F_{\text{b,1}} = \frac{N_1}{N_{\text{ants,total}} - N_1}$ and $F_{\text{b,2}} = \frac{N_{\text{ants,total}} - N_2}{N_2}$. Since in Case 1 both $F_{\text{b,1}}$ and $F_{\text{b,2}}$ are negative, and $F(t) > 0$ always holds by definition. Therefore, in Case 1, only M1 is possible, and the other modes cannot occur. Using a similar reasoning, each of the 4 cases can be summarized in Table A.1.

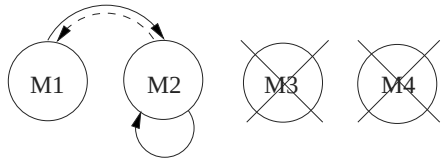
The processes of updating pheromone for Cases $1-4$ is illustrated in Figure A.2. In particular, Case 4 has three sub-cases:

- Subcase 4.a: $N_{\text{ants,total}} \leqslant N_1 + N_2$
- Subcase 4.b: $N_{\text{ants,total}} > N_1 + N_2$ and $P \leqslant \dfrac{(f_1 - f_2)(N_{\text{ants,total}} - N_2)}{N_{\text{ants,total}} - N_1 - N_2}$
- Subcase 4.c : $N_{\text{ants,total}} > N_1 + N_2$ and $P > \dfrac{(f_1 - f_2)(N_{\text{ants,total}} - N_2)}{N_{\text{ants,total}} - N_1 - N_2}$

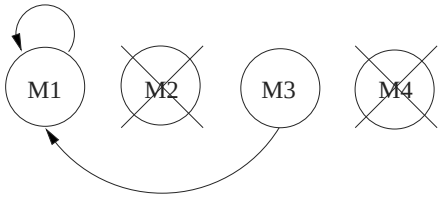
This will be explained in more detail in Section A.3.



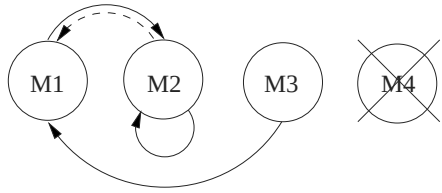
(a) Mode transitions in Case 1



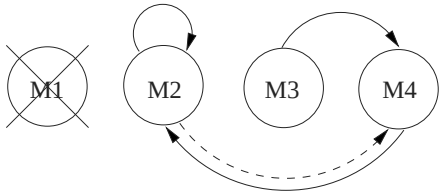
(b) Mode transitions in Case 2



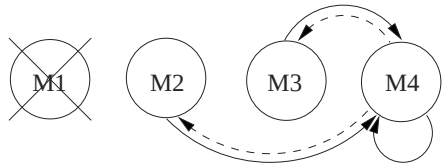
(c) Mode transitions in Case 3



(d) Mode transitions of Subcase 4.a



(e) Mode transitions of Subcase 4.b



(f) Mode transitions of Subcase 4.c

Figure A.2: Mode transitions in four different cases. The transitions indicated by a dashed line can occur only once.

A.2 Convergence Properties of F

In this section, the properties of F are investigated. As given in Section A.1.2, $F_{\mathrm{b},1}$ and $F_{\mathrm{b},2}$ represent the mode boundaries of F. If the value of F becomes larger or smaller than $F_{\mathrm{b},1}$ or $F_{\mathrm{b},2}$, then a transition will occur from one mode to another.

Lemma A.2 *In M*1 *and M*3*, F is a monotonically increasing function.*

Proof: In M1, the pheromone update equations are formulated as:

$$\begin{aligned}
\tau_1(t+1) &= (1-\rho_{\mathrm{evap}})\tau_1(t) + N_1(t) f_1 \\
&= (1-\rho_{\mathrm{evap}})\tau_1(t) + \frac{\tau_1(t)}{\tau_1(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_1 \\
\tau_2(t+1) &= (1-\rho_{\mathrm{evap}})\tau_2(t) + n_2(t) f_2 \\
&= (1-\rho_{\mathrm{evap}})\tau_2(t) + \frac{\tau_2(t)}{\tau_1(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_2
\end{aligned}$$

The value $F(t+1)$ can thus be written as:

$$\begin{aligned}
F(t+1) = \frac{\tau_1(t+1)}{\tau_2(t+1)} &= \frac{(1-\rho_{\mathrm{evap}})\tau_1(t) + \frac{\tau_1(t)}{\tau_1(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_1}{(1-\rho_{\mathrm{evap}})\tau_2(t) + \frac{\tau_2(t)}{\tau_1(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_2} \\
&= \frac{(1-\rho_{\mathrm{evap}})\tau_2(t)F(t) + \frac{\tau_2(t)F(t)}{\tau_2(t)F(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_1}{(1-\rho_{\mathrm{evap}})\tau_2(t) + \frac{\tau_2(t)}{\tau_2(t)F(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_2} \\
&= F(t) \frac{(1-\rho_{\mathrm{evap}})\tau_2(t) + \frac{1}{F(t)+1} N_{\mathrm{ants,total}} f_1}{(1-\rho_{\mathrm{evap}})\tau_2(t) + \frac{1}{F(t)+1} N_{\mathrm{ants,total}} f_2}
\end{aligned}$$

Because of Assumption A.1, i.e., $f_1 > f_2$, we can conclude that $F(t+1) > F(t)$, which means that $F(\cdot)$ is a strictly monotonically increasing function in M1.

In M3, the pheromone update equations are formulated as:

$$\begin{aligned}
\tau_1(t+1) &= (1-\rho_{\mathrm{evap}})\tau_1(t) + n_1(t) f_1 \\
&= (1-\rho_{\mathrm{evap}})\tau_1(t) + \frac{\tau_1(t)}{\tau_1(t)+\tau_2(t)} N_{\mathrm{ants,total}} f_1 \\
\tau_2(t+1) &= (1-\rho_{\mathrm{evap}})\tau_2(t) + n_2(t) f_2 - P(n_2(t) - N_2) \\
&= (1-\rho_{\mathrm{evap}})\tau_2(t) + n_2(t)(f_2 - P) + P N_2 \\
&= (1-\rho_{\mathrm{evap}})\tau_2(t) + \frac{\tau_2(t)}{\tau_1(t)+\tau_2(t)} N_{\mathrm{ants,total}} (f_2 - P) + P N_2
\end{aligned}$$

Hence, $F(t+1)$ can be written as:

$$F(t+1) = F(t)\frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_1}{F(t)+1}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}}(f_2 - P)}{F(t)+1} + PN_2}$$

Since we want to prove that $F(t+1) > F(t)$, it should hold that

$$\frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_1}{F(t)+1}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}}(f_2 - P)}{F(t)+1} + PN_2} > 1$$

To satisfy the inequality above, we need

$$F(t) < \frac{N_{\text{ants,total}}(f_1 - f_2) + P(N_{\text{ants,total}} - N_2)}{PN_2}, \tag{A.15}$$

Because of Assumption A.1, the right-hand side of (A.15) is larger than $F_{\text{b},2}$. Therefore, in M3, (A.15) always holds, and F is a strictly monotonically increasing function in M3. □

Define

$$\begin{aligned} F_{\text{equ},1} =& \frac{PN_1}{N_{\text{ants,total}}(f_2 - f_1) + P(N_{\text{ants,total}} - N_1)} \\ F_{\text{equ},2} =& \frac{1}{2PN_2}\Big(N_{\text{ants,total}}(f_1 - f_2) + P(N_1 - N_2) + \\ & \sqrt{\big(N_{\text{ants,total}}(f_1 - f_2) + P(N_1 - N_2)\big)^2 + 4P^2 N_1 N_2}\Big) \end{aligned}$$

Lemma A.3 *If $F_{\text{equ},1} \geqslant \max(F_{\text{b},1}, F_{\text{b},2})$, then $F_{\text{equ},1}$ is the only equilibrium point of F in M2. In M2, when $F(t) > F_{\text{equ},1}$, then $F(t+1) < F(t)$, and when $F(t) < F_{\text{equ},1}$, then $F(t+1) > F(t)$. If $F_{\text{b},1} < F_{\text{equ},2} < F_{\text{b},2}$, then $F_{\text{equ},2}$ is the only equilibrium point of F in M4. In M4, when $F(t) > F_{\text{equ},2}$, then $F(t+1) < F(t)$, and when $F(t) < F_{\text{equ},2}$, then $F(t+1) > F(t)$.*

Proof: In M2, the pheromone update equations are given by:

$$\begin{aligned} \tau_1(t+1) &= (1-\rho_{\text{evap}})\tau_1(t) + n_1(t) f_1 - P(n_1(t) - N_1) \\ &= (1-\rho_{\text{evap}})\tau_1(t) + n_1(t)(f_1 - P) + N_1 P \\ &= (1-\rho_{\text{evap}})\tau_1(t) + \frac{\tau_1(t)}{\tau_1(t)+\tau_2(t)} N_{\text{ants,total}}(f_1 - P) + N_1 P \\ \tau_2(t+1) &= (1-\rho_{\text{evap}})\tau_2(t) + n_2(t) f_2 \\ &= (1-\rho_{\text{evap}})\tau_2(t) + \frac{\tau_2(t)}{\tau_1(t)+\tau_2(t)} N_{\text{ants,total}} f_2 \end{aligned}$$

The value $F(t+1)$ can be written as:

$$F(t+1) = F(t)\frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}}(f_1-P)}{F(t)+1} + \dfrac{N_1 P}{F(t)}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} \tag{A.16}$$

Since we want to find the equilibrium point, we let $F(t+1) = F(t) = F_{\text{e}}$. Therefore, we have:

$$\frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}}(f_1-P)}{F_{\text{e}}+1} + \dfrac{N_1 P}{F_{\text{e}}}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F_{\text{e}}+1}} = 1 \tag{A.17}$$

This yields

$$F_{\text{e}} = \frac{N_1 P}{N_{\text{ants,total}}(f_2-f_1) + P(N_{\text{ants,total}} - N_1)} \tag{A.18}$$

This is the equilibrium point $F_{\text{equ,1}}$. Furthermore, if $F(t) < F_{\text{equ,1}}$, the factor that multiples $F(t)$ in (A.16) is larger than 1, which means that $F(t+1) > F(t)$, and if $F(t) > F_{\text{equ,1}}$, that factor is smaller than 1, which means that $F(t+1) < F(t)$.

Similarly, we can prove that $F_{\text{equ,2}}$ is the only equilibrium point in M4 . When $F(t) < F_{\text{equ,1}}$, we have $F(t+1) > F(t)$, and when $F(t) > F_{\text{equ,1}}$, we have $F(t+1) < F(t)$. □

Assumption A.3

$$P > \frac{N_{\text{ants,total}}(f_1-f_2)}{N_{\text{ants,total}} - N_1} \tag{A.19}$$

With Assumption A.3, it is easy to verify $F_{\text{equ,1}} > 0$, and $F_{\text{equ,2}} > 0$ always holds. Therefore, if $F(t) < F_{\text{equ,1}}$ when $F(t)$ is in M2, or if $F(t) < F_{\text{equ,2}}$ when $F(t)$ is in M4, $F(\cdot)$ is a monotonically increasing function, while if $F(t) > F_{\text{equ,1}}$ when $F(t)$ is in M2, or if $F(t) > F_{\text{equ,2}}$ when $F(t)$ is in M4, $F(\cdot)$ is a monotonically decreasing function. In the other words, F always moves towards an equilibrium point when it stays in M2 or M4. However, Lemma A.3 does not guarantee that F will converge to either $F_{\text{equ,1}}$ or $F_{\text{equ,2}}$ as it could still oscillate around $F_{\text{equ,1}}$ or $F_{\text{equ,2}}$. Therefore, another lemma is introduced.

Lemma A.4 *In M2,* $\lim_{t\to\infty} |F(t) - F_{\text{equ,1}}| = 0$, *and in M4,* $\lim_{t\to\infty} |F(t) - F_{\text{equ,2}}| = 0$.

Proof: We first reformulate (A.18) as follows:

$$\begin{aligned} & N_1 P = F_{\text{equ,1}}\left(N_{\text{ants,total}}(f_2-f_1) + P(N_{\text{ants,total}} - N_1)\right) \\ \Rightarrow\quad & N_1 P = F_{\text{equ,1}} N_{\text{ants,total}} f_2 - F_{\text{equ,1}} N_{\text{ants,total}} f_1 + F_{\text{equ,1}} N_{\text{ants,total}} P - F_{\text{equ,1}} N_1 P \\ \Rightarrow\quad & F_{\text{equ,1}} N_{\text{ants,total}} f_2 = F_{\text{equ,1}} N_{\text{ants,total}}(f_1-P) + (F_{\text{equ,1}}+1) N_1 P \end{aligned} \tag{A.20}$$

Then,

$$|F(t+1) - F_{\text{equ,1}}|$$

$$
\begin{aligned}
&= \left| F(t) \frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}}(f_1 - P)}{F(t)+1} + \dfrac{N_1 P}{F(t)}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} - F_{\text{equ,1}} \right| \\
&= \left| \frac{F(t)(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{F(t) N_{\text{ants,total}}(f_1 - P)}{F(t)+1} + N_1 P - F_{\text{equ,1}} \left((1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1} \right)}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} \right| \\
&= \left| \frac{(F(t) - F_{\text{equ,1}})(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{F(t) N_{\text{ants,total}}(f_1 - P) - \overbrace{F_{\text{equ,1}} N_{\text{ants,total}} f_2}^{\text{substituted using (A.20)}}}{F(t)+1} + N_1 P}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} \right| \\
&= \left| \frac{(F(t) - F_{\text{equ,1}})(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{(F(t) - F_{\text{equ,1}}) \left(N_{\text{ants,total}} f_1 - N_{\text{ants,total}} P + N_1 P \right)}{F(t)+1}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} \right| \\
&= |F(t) - F_{\text{equ,1}}| \cdot \left| \frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{\left(N_{\text{ants,total}} f_1 - N_{\text{ants,total}} P + N_1 P \right)}{F(t)+1}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} \right| \qquad \text{(A.21)}
\end{aligned}
$$

From Assumption A.3, we can derive from that $N_{\text{ants,total}}(f_2 - f_1) + P(N_{\text{ants,total}} - N_1) > 0$, which means that $N_{\text{ants,total}} f_1 - N_{\text{ants,total}} P + N_1 P < N_{\text{ants,total}} f_2$. Recall from Assumption A.2 $N_{\text{ants,total}} f_1 - N_{\text{ants,total}} P + N_1 P > 0$. Therefore, we can conclude that

$$
0 < \left| \frac{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{\left(N_{\text{ants,total}} f_1 - N_{\text{ants,total}} P + N_1 P \right)}{F(t)+1}}{(1-\rho_{\text{evap}})\tau_2(t) + \dfrac{N_{\text{ants,total}} f_2}{F(t)+1}} \right| < 1
$$

As a result, (A.21) is a contraction to 0, which implies $\lim_{t \to \infty} |F(t+1) - F_{\text{equ,1}}| = 0$.

In a similar way, we can also prove that $\lim_{t \to \infty} |F(t+1) - F_{\text{equ,2}}| = 0$. □

From Lemma A.4, F will asymptotically converge to these equilibrium points. However, Lemma A.4 can only be applied if F always stays in M2 or M4. It is possible that F jumps out of M2 or M4 due to the mode transition. In such case, the convergence is still not guaranteed.

Lemma A.5 *In all four modes, if F transits from $M^\dagger$ to M^*, where $M^\dagger$ denotes a mode without an equilibrium point, and M^* denotes a mode with an equilibrium point, F will stay in M^*.*

Proof: When $N_{\text{ants,total}} \leqslant N_1 + N_2$ as in Subcase 4.a, the equilibrium point $F_{\text{equ,1}}$ is in M2, and F can only transit from M1 to M2. We suppose that $F(t_0)$ is in M1, in which $F(t_0) < F_{\text{b,1}}$, and $F(t_0+1)$ is in M2, in which $F(t_0+1) > F_{\text{b,1}}$. From the contraction in (A.21), we know that F always moves towards $F_{\text{equ,1}}$ in M2. If we can prove that $|F(t_0+1) - F_{\text{equ,1}}| < F_{\text{equ,1}} - F_{\text{b,1}}$, then

we have proved that $F(t)$ will stay in M2 from iteration step t_0. Because $F(t_0+1) > F_{\mathrm{b},1}$, it is clear that $F_{\mathrm{b},1} - F_{\mathrm{equ},1} < F(t_0+1) - F_{\mathrm{equ},1}$. We only need to prove $F(t_0+1) - F_{\mathrm{equ},1} < F_{\mathrm{equ},1} - F_{\mathrm{b},1}$. Since $F(t_0)$ is in M1, $F(t_0+1)$ is calculated by the equations of M1:

$$
\begin{aligned}
&(F(t_0+1) - F_{\mathrm{equ},1}) - (F_{\mathrm{equ},1} - F_{\mathrm{b},1}) \\
=&F(t_0)\frac{(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_1}{F(t_0)+1}}{(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_2}{F(t_0)+1}} + F_{\mathrm{b},1} - 2F_{\mathrm{equ},1} \\
<&F_{\mathrm{b},1}\frac{(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_1}{F(t_0)+1}}{(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_2}{F(t_0)+1}} + F_{\mathrm{b},1} - 2F_{\mathrm{equ},1} \\
=&F_{\mathrm{b},1}\frac{2(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}}(f_1+f_2)}{F(t_0)+1}}{(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_2}{F(t_0)+1}} - 2F_{\mathrm{equ},1} \\
=&\frac{N_1}{N_{\mathrm{ants,total}} - N_1} \cdot \frac{2(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}}(f_1+f_2)}{F(t_0)+1}}{(1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_2}{F(t_0)+1}} - \frac{2PN_1}{N_{\mathrm{ants,total}}(f_2-f_1) + P(N_{\mathrm{ants,total}} - N_1)} \\
=&\frac{N_1}{(N_{\mathrm{ants,total}} - N_1)\left((1-\rho_{\mathrm{evap}})\tau_2(t_0) + \dfrac{N_{\mathrm{ants,total}} f_2}{F(t_0)+1}\right)} \cdot \frac{1}{\left(N_{\mathrm{ants,total}}(f_2-f_1) + P(N_{\mathrm{ants,total}} - N_1)\right)} \cdot \\
&\left(2(1-\rho_{\mathrm{evap}})\tau_2(t_0) N_{\mathrm{ants,total}}(f_2-f_1) + \frac{N_{\mathrm{ants,total}}}{F(t_0)+1}(f_2-f_1)\big(N_{\mathrm{ants,total}}(f_1+f_2) - P(N_{\mathrm{ants,total}} - N_1)\big)\right)
\end{aligned}
$$

By using $f_1 > f_2$, as well as Assumption A.2.1, we can prove that $(F(t_0+1) - F_{\mathrm{equ},1}) - (F_{\mathrm{equ},1} - F_{\mathrm{b},1}) < 0$. As a conclusion, we have $|F(t_0+1) - F_{\mathrm{equ},1}| < F_{\mathrm{equ},1} - F_{\mathrm{b},1}$.

We can also use the similar method to prove that when $N_{\mathrm{ants,total}} > N_1 + N_2$, if $F(\cdot)$ transits into M2 or M4 , it will stay in those modes. □

Lemma A.5 shows that F will not jump out of M2 or M4 when it enters these modes. More specifically, as shown in Figures A.2(b) and A.2(d), when F transits from M1 to M2, F will not go back to M1. The case that F may transit from M2 to M1 (shown by the dashed line) can only occur when F is initialized in M2, and such a transition can only occur once. Similarly, in Figure A.2(e), when F transits from M4 to M2, F will not go back to M4, and in Figure A.2(f), when F transits from M2 and M3 to M4, F will not go back to neither M2 nor M3.

A.3 Convergence of the pheromone levels

Proposition A.1 *In Case* 1*, the pheromone levels on both arcs asymptotically converge to a finite value.*

Proof: In Case 1, F will only stay in M1. According to Lemma A.2, F is a monotonically increasing function in M1, so the value of $F(t)$ will monotonically converge when $t \to \infty$.

Let[1] $F' = \lim_{t\to\infty} F(t)$. According to (A.12),

$$\lim_{t\to\infty} n_1(t) = \lim_{t\to\infty} \frac{F(t)}{F(t)+1} \cdot N_{\text{ants,total}} = \frac{1/F'}{1/F'+1} \cdot N_{\text{ants,total}},$$
$$\lim_{t\to\infty} n_2(t) = \lim_{t\to\infty} \frac{1}{F(t)+1} \cdot N_{\text{ants,total}} = \frac{1}{F'+1} \cdot N_{\text{ants,total}}.$$

Therefore, the numbers of ants $n_1(t)$ and $n_2(t)$ also converge.

Given a difference equation $x(t+1) = ax(t) + b(t)$, with $0 < a < 1$, if $\lim_{t\to\infty} b(t) = B$, then:

$$\forall \varepsilon > 0,\ \exists T : B - \varepsilon < b(t) < B + \varepsilon, \forall\, t > T.$$

Therefore,

$$ax(t) + B - \varepsilon < ax(t) + b(t) < ax(t) + B + \varepsilon, \forall\, t > T.$$

This is equivalent to

$$ax(t) + B - \varepsilon < x(t+1) < ax(t) + B + \varepsilon, \forall\, t > T.$$

From $x(t+1) < ax(t) + B + \varepsilon$, we can conclude that

$$x(t+1) < a^t x(0) + (a^{t-1} + a^{t-2} + \cdots + 1)(B + \varepsilon)$$

Now select $T' > T$ such that

$$a^t x(0) < \frac{\varepsilon}{1-a},\ \ \forall\, t > T'$$

Since $0 < a < 1$, such a T' always exists. Then for all $t > T'$, it holds that

$$\begin{aligned} x(t+1) &< \frac{\varepsilon}{1-a} + \frac{1}{1-a}(B+\varepsilon) \\ &< \frac{B}{1-a} + \frac{2\varepsilon}{1-a} \end{aligned}$$

Moreover, because $a > 0$, it holds that

$$x(t+1) > \frac{1}{1-a}(B-\varepsilon) > \frac{B}{1-a} - \frac{2\varepsilon}{1-a}$$

Defining $\varepsilon' = \dfrac{2\varepsilon}{1-a}$, we find

$$\forall \varepsilon' > 0,\ \exists T' : \frac{B}{1-a} - \varepsilon' < x(t+1) < \frac{B}{1-a} + \varepsilon', \forall\, t > T'.$$

Hence,

$$\lim_{t\to\infty} x(t+1) = \frac{B}{1-a} \tag{A.22}$$

[1] F' can be ∞. In that case, $\lim_{t\to\infty} n_1(t) = N_{\text{ants,total}}$, and $\lim_{t\to\infty} n_2(t) = 0$.

Using (A.22) for (A.3), it is proven that pheromone levels $\tau_1(t)$ and $\tau_2(t)$ converge. □

Proposition A.2 *In Case 2, the pheromone levels on both arcs asymptotically converge.*

Proof: We first prove the mode transition in Figure A.2(b). In Case 2, M3 and M4 cannot be reached. Due to Lemma A.2, if F is initialized in M1, it will keep increasing until reaching $F_{\mathrm{b},1}$, and then it will transit to M2. If F is initialized in M2, it may transit from M2 to M1. However, Lemma A.5 proves that after F transits from M1 to M2, it will stay in M2, because M2 has an equilibrium point $F_{\mathrm{equ},1}$, while M1 has no equilibrium point. In this way, the process described by Figure A.2(b) is proved.

Since F finally stays in M2 , it will converge to $F_{\mathrm{equ},1}$, as stated in Lemma A.4. According to (A.12),

$$\lim_{t\to\infty} n_1(t) = \lim_{t\to\infty} \frac{F(t)}{F(t)+1} \cdot N_{\mathrm{ants,total}} = \frac{F_{\mathrm{equ},1}}{F_{\mathrm{equ},1}+1} \cdot N_{\mathrm{ants,total}} ,$$
$$\lim_{t\to\infty} n_2(t) = \lim_{t\to\infty} \frac{1}{F(t)+1} \cdot N_{\mathrm{ants,total}} = \frac{1}{F_{\mathrm{equ},1}+1} \cdot N_{\mathrm{ants,total}} .$$

Therefore, the numbers of ants $n_1(t)$ and $n_2(t)$ also converge, which results in convergence of the pheromone levels $\tau_1(t)$ and $\tau_2(t)$. □

Proposition A.3 *In Case 3, the pheromone levels on both arcs asymptotically converge.*

Proof: Similar to Propostion A.2. □

Lemma A.6 *In Case 4, if* $N_1 < N_{\mathrm{ants,total}} \leqslant N_1 + N_2$, *then* $F_{\mathrm{b},1} \geqslant F_{\mathrm{b},2}$, *and if* $N_{\mathrm{ants,total}} > N_1 + N_2$, *then* $F_{\mathrm{b},1} < F_{\mathrm{b},2}$.

Proof: We have

$$\begin{aligned} F_{\mathrm{b},1} - F_{\mathrm{b},2} &= \frac{N_1}{N_{\mathrm{ants,total}} - N_1} - \frac{N_{\mathrm{ants,total}} - N_2}{N_2} \\ &= \frac{N_1 N_2 - (N_{\mathrm{ants,total}} - N_1)(N_{\mathrm{ants,total}} - N_2)}{(N_{\mathrm{ants,total}} - N_1) N_2} \\ &= \frac{N_{\mathrm{ants,total}}(N_1 + N_2 - N_{\mathrm{ants,total}})}{(N_{\mathrm{ants,total}} - N_1) N_2} \end{aligned}$$

If $N_1 < N_{\mathrm{ants,total}} \leqslant N_1 + N_2$, then $F_{\mathrm{b},1} \geqslant F_{\mathrm{b},2}$, and if $N_{\mathrm{ants,total}} < N_1 + N_2$, then $F_{\mathrm{b},1} > F_{\mathrm{b},2}$. □

Proposition A.4 *In Case 4, the pheromone levels on both arcs asymptotically converge.*

Proof: In Subcase 4.a, we know that $F_{\mathrm{b},1} \geqslant F_{\mathrm{b},2}$ from Lemma A.6. The four modes M1 –M4 can be further described as:

$$\begin{aligned} &\mathrm{M1} : F_{\mathrm{b},2} \leqslant F(t) \leqslant F_{\mathrm{b},1} , \\ &\mathrm{M2} : F(t) > F_{\mathrm{b},1} , \\ &\mathrm{M3} : 0 < F(t) < F_{\mathrm{b},2} , \\ &\mathrm{M4} : \mathrm{NR}. \end{aligned}$$

As proved in Lemma A.2, F monotonically increases in both M1 and M3. Therefore, if F is initialized in M3, it will eventually transit to M1, and if F is initialized in M1, it will eventually transit to M2. Because $f_1 > f_2$ according to Assumption A.1, $F_{\text{equ},1} > F_{\text{b},1}$ always holds, which means that $F_{\text{equ},1}$ is always located in the range of M2 due to Lemma A.3. Therefore, similarly to Case 2, if F is initialized in M2, it may transit from M1 to M2, but after F transits from M1 to M2, it will stay in M2 according to Lemma A.5. Moreover, F will finally converge to $F_{\text{equ},1}$, which proves the mode transitions of Figure A.2(d).

In Subcase 4.b and Subcase 4.c, we know that $F_{\text{b},1} < F_{\text{b},2}$ from Lemma A.6. The four modes M1–M4 can be further described as:

$$\begin{aligned}
&\text{M1} : \text{NR} , \\
&\text{M2} : F(t) \geqslant F_{\text{b},2} , \\
&\text{M3} : 0 < F(t) \leqslant F_{\text{b},1} , \\
&\text{M4} : F_{\text{b},1} < F(t) < F_{\text{b},2} .
\end{aligned}$$

In Subcase 4.b, we have $P \leqslant \dfrac{(f_1 - f_2)(N_{\text{ants,total}} - N_2)}{N_{\text{ants,total}} - N_1 - N_2}$, and one can prove that $F_{\text{equ},1} > F_{\text{b},2}$ and $F_{\text{equ},2} > F_{\text{b},2}$. As a result, $F_{\text{equ},1}$ is in M2, while $F_{\text{equ},2}$ is outside the range of M4. Since F is a monotonically increasing function in M4, we can use a method similar to that of Subcase 4.a to prove the mode transitions of Figure A.2(e), where F will also converge to $F_{\text{equ},1}$ in M2.

In Subcase 4.c, we have $P > \dfrac{(f_1 - f_2)(N_{\text{ants,total}} - N_2)}{N_{\text{ants,total}} - N_1 - N_2}$, and one can prove that $F_{\text{equ},1} < F_{\text{b},2}$, and $F_{\text{b},1} < F_{\text{equ},2} < F_{\text{b},2}$. As a result, $F_{\text{equ},1}$ is outside the range of M2, while $F_{\text{equ},2}$ is in M4. Since F is a monotonically decreasing function in M2, we can also prove the mode transitions of Figure A.2(f) similar to Subcase 4.a, where F will converge to $F_{\text{equ},2}$ in M4.

Since all of the three subcases of Case 4 lead to convergence of F, we can prove that in Case 4 the numbers of ants $n_1(t)$ and $n_2(t)$ converge, and accordingly the pheromone levels also converge. □

From Propositions A.1–A.4, Definition A.1 is satisfied. In conclusion, the convergence of ACO-SP in a graph with two arcs is proven.

Appendix B

The Basic METANET Model

For the sake of completeness, the basic METANET traffic flow model used in thesis is provided in this appendix. This appendix is based on Chapter 3 of [70] [1].

The METANET model was originally developed by Messmer and Papageorgiou [98]. It is a deterministic, discrete-time, discrete-space, and macroscopic model. The METANET model can operate in two modes: the destination-independent and the destination-dependent mode. The destination-dependent mode is useful for networks that have multiple destinations and the possibility for route choice. We first present the equations for the destination-independent mode, and next we give the extensions necessary to represent destination-dependent traffic. For the full description of METANET we refer to the literature [86, 98].

B.1 Link equations

The METANET model represents a network as a directed graph with the links (indicated by the index m) corresponding to freeway stretches. Each freeway link has uniform characteristics, i.e., no on-ramps or off-ramps and no major changes in geometry. Where major changes occur in the characteristics of the link or in the road geometry (e.g., on-ramp or an off-ramp), a node is placed. Each link m is divided into N_m segments (indicated by the index i) of length L_m (typically 500-1000 m, see also Figure B.1). Each segment i of link m is characterized by three quantities:

- *traffic density* $\rho_{m,i}(k)$ (veh/km/lane),
- *space-mean speed* $v_{m,i}(k)$ (km/h),
- *traffic volume* or *outflow* $q_{m,i}(k)$ (veh/h),

where k indicates the time instant $t = kT$, and T is the time step used for the simulation of the traffic flow (typically $T = 10$ s). For stability, the segment length and the simulation time step should satisfy for every link m

$$L_m > v_{\text{free},m} T \ , \tag{B.1}$$

where $v_{\text{free},m}$ is the average speed that drivers assume if traffic is freely flowing.

[1] We would like to thank Dr. András Hegyi for the permission to include this material here, and for sharing the source files of his chapter with us.

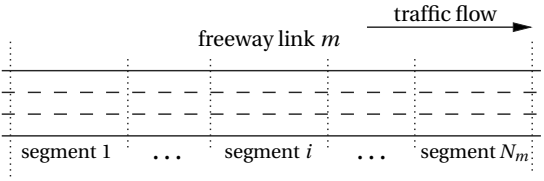


Figure B.1: In the METANET model a freeway link is divided into segments.

The outflow of each segment is equal to the density multiplied by the mean speed and the number of lanes on that segment (denoted by λ_m):

$$q_{m,i}(k) = \rho_{m,i}(k)\, v_{m,i}(k)\, \lambda_m \ . \tag{B.2}$$

The density of a segment equals the previous density plus the inflow from the upstream segment, minus the outflow of the segment itself (conservation of vehicles):

$$\rho_{m,i}(k+1) = \rho_{m,i}(k) + \frac{T}{L_m \lambda_m}\big(q_{m,i-1}(k) - q_{m,i}(k)\big) \ . \tag{B.3}$$

While equations (B.2) and (B.3) are based on physical principles, the equations that describe the speed dynamics and the relation between density and the desired speed are heuristic. In the METANET model the mean speed at the simulation step $k+1$ is taken to be the mean speed at time step k plus a *relaxation term* that expresses that the drivers try to achieve a desired speed $V(\rho)$, a *convection term* that expresses the speed increase (or decrease) caused by the inflow of vehicles, and an *anticipation term* that expresses the speed decrease (increase) as drivers experience a density increase (decrease) downstream:

$$\begin{aligned} v_{m,i}(k+1) = v_{m,i}(k) &+ \frac{T}{\tau}\Big(V\big(\rho_{m,i}(k)\big) - v_{m,i}(k)\Big) + \\ &\frac{T}{L_m} v_{m,i}(k)\big(v_{m,i-1}(k) - v_{m,i}(k)\big) - \frac{\eta T}{\tau L_m}\frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa} \ , \end{aligned} \tag{B.4}$$

where τ, η[2], and κ are model parameters, and with

$$V\big(\rho_{m,i}(k)\big) = v_{\text{free},m} \exp\left[-\frac{1}{a_m}\left(\frac{\rho_{m,i}(k)}{\rho_{\text{crit},m}}\right)^{a_m}\right] \ , \tag{B.5}$$

with a_m a model parameter, and where the free-flow speed $v_{\text{free},m}$ is the average speed that drivers assume if traffic is freely flowing, and the critical density $\rho_{\text{crit},m}$ is the density at which the traffic flow is maximal on a homogeneous freeway.

Origins are modeled with a simple queue model. The length $w_o(k)$ of the queue at origin o equals the previous queue length plus the demand $d_o(k)$, minus the outflow $q_o(k)$:

$$w_o(k+1) = w_o(k) + T\big(d_o(k) - q_o(k)\big) \ .$$

The outflow of origin o depends on the traffic conditions on the main stream freeway and,

[2] In the original METANET model this parameter is denoted by ν (nu), but because of the small typographical difference with v (speed) we prefer to use η.

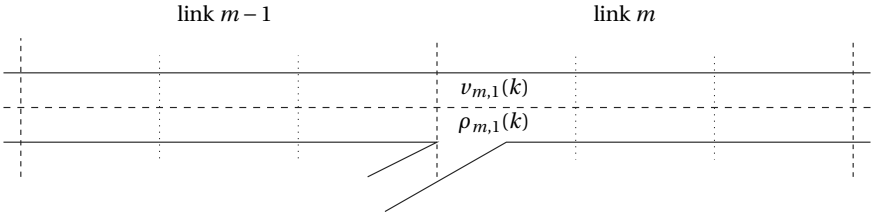


Figure B.2: When there is an on-ramp connected to the freeway the speed $v_{m,1}(k)$ in the first segment of link m is reduced by merging phenomena according to (B.7).

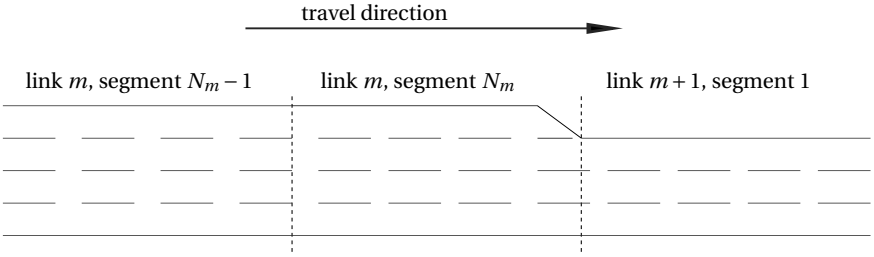


Figure B.3: When there is a lane drop the speed $v_{m,N_m}(k)$ in the last segment of link m is reduced by merging phenomena according to (B.8).

for a metered on-ramp, on the ramp metering rate[3] $r_o(k)$, where $r_o(k) \in [0,1]$. The ramp flow $q_o(k)$ is the minimum of three quantities:

- the available traffic at simulation step k (queue plus demand),
- the maximal flow allowed by the metering rate,
- and the maximal flow that could enter the freeway because of the main-stream conditions.

So,

$$q_o(k) = \min\left[d_o(k) + \frac{w_o(k)}{T},\, C_o r_o(k),\, C_o\left(\frac{\rho_{\max,m} - \rho_{m,1}(k)}{\rho_{\max,m} - \rho_{\text{crit},m}}\right)\right], \tag{B.6}$$

where C_o is the on-ramp capacity (veh/h) under free-flow conditions, the global parameter $\rho_{\max,m}$ (veh/km/lane) is the maximum density of a segment (also called jam density), and m is the index of the link to which the on-ramp is connected.

In order to account for the speed drop caused by merging phenomena, if there is an on-ramp, then the term

$$-\frac{\delta T q_o(k) v_{m,1}(k)}{L_m \lambda_m (\rho_{m,1}(k) + \kappa)} \tag{B.7}$$

is added to (B.4), where $v_{m,1}(k)$ and $\rho_{m,1}(k)$ are the speed and density of the segment that the on-ramp is connected to, as shown in Figure B.2, and δ is a model parameter.

[3] For an unmetered on-ramp we also can use (B.6) by setting $r_o(k) \equiv 1$.

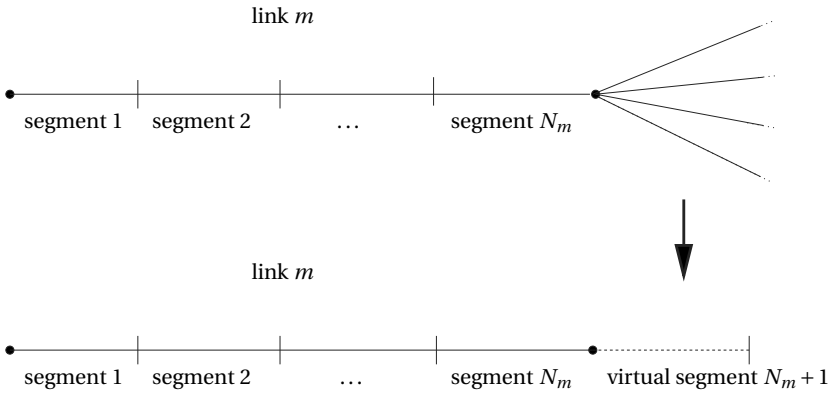


Figure B.4: A node with one entering link m and several leaving links. The densities in the first segments of the leaving links are aggregated in the virtual downstream density $\rho_{m,N_m+1}(k)$ according to (B.9).

When there is a lane drop as shown in Figure B.3, the speed reduction due to weaving phenomena,

$$-\frac{\phi T \Delta\lambda_m \rho_{m,N_m}(k)\, v^2_{m,N_m}(k)}{L_m \lambda_m \rho_{\text{crit},m}} \quad , \tag{B.8}$$

is added to (B.4), where $\Delta\lambda_m = \lambda_m - \lambda_{m+1}$ is the number of lanes being dropped, and ϕ is a model parameter.

B.2 Node equations

The coupling equations to connect links are as follows. Every time there is a major change in the link parameters or there is a junction or a bifurcation, a node is placed between the links. This node provides the incoming links with a downstream density (or a virtual downstream density when there are more leaving links), and the leaving links with an inflow and an upstream speed (or a virtual upstream speed when there are more entering links). The flow that enters node n is distributed among the leaving links according to

$$\begin{aligned} Q_n(k) &= \sum_{\mu \in \mathscr{I}_n} q_{\mu,N_\mu}(k) \quad , \\ q_{m,0}(k) &= \beta_{n,m}(k) Q_n(k) \quad , \end{aligned}$$

where $Q_n(k)$ is the total flow that enters node n at simulation step k, $\mathscr{I}_n$ is the set of links that enter node n, $\beta_{n,m}(k)$ are the turning rates (the fraction of the total flow through node n that leaves via link m), and $q_{m,0}(k)$ is the flow that leaves node n via link m, where link m is one of the links leaving node n.

When node n has more than one leaving link as shown in Figure B.4, the virtual down-

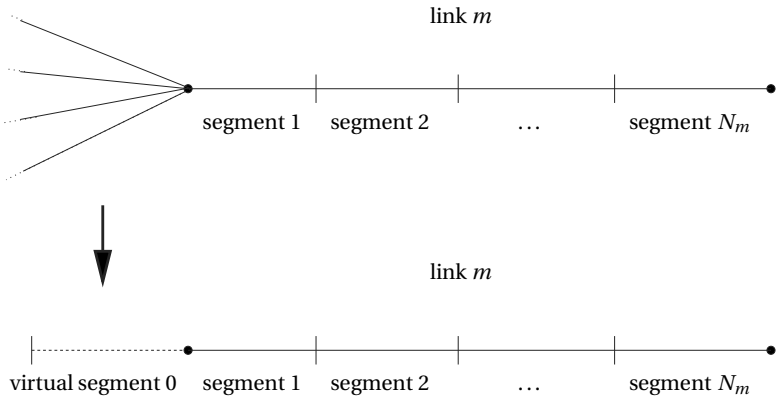


Figure B.5: A node with one leaving link m and several entering links. The speeds in the last segments of the entering links are aggregated in the virtual upstream speed $v_{m,0}(k)$ according to (B.10).

stream density $\rho_{m,N_m+1}(k)$ of entering link m is given by

$$\rho_{m,N_m+1}(k) = \frac{\sum_{\mu \in \mathscr{O}_n} \rho_{\mu,1}^2(k)}{\sum_{\mu \in \mathscr{O}_n} \rho_{\mu,1}(k)} \ , \tag{B.9}$$

where $\mathscr{O}_n$ is the set of links leaving node n.

When node n has more than one entering link as shown in Figure B.5, the virtual upstream speed $v_{m,0}(k)$ of leaving link m is given by

$$v_{m,0}(k) = \frac{\sum_{\mu \in \mathscr{I}_n} v_{\mu,N_\mu}(k) q_{\mu,N_\mu}(k)}{\sum_{\mu \in \mathscr{I}_n} q_{\mu,N_\mu}(k)} \ . \tag{B.10}$$

B.3 Boundary conditions

Boundary conditions need to be defined for the entries and exits of the traffic network. As in METANET the state of a segment also depends on the upstream speed, the outflow of the upstream node, and the downstream density, we need to define the upstream speed and the inflow for the entries of the network, and the downstream density for the exits of the network. These boundary conditions can be user-specified or a default value can be assumed. We already presented the boundary conditions for the traffic demand at on-ramps, now we present the boundary conditions for the upstream speed and the downstream density.

B.3.1 Upstream speed

When there is a main-stream origin entering node n, the virtual speed $v_\mu(k)$ of the origin can be user-specified, where μ is the index of the origin. If $v_\mu(k)$ is not specified, then it is often

set equal to the speed of the first segment of the leaving link m

$$v_\mu(k) = v_{m,1}(k) \ .$$

B.3.2 Downstream density

Similarly, the virtual downstream $\rho_{m,N_m+1}(k)$ density for the entering link at a node that is connected to a destination, is calculated as follows. First, the user can specify a destination density scenario $\rho_\mu(k)$ where μ is the index of the destination link. Alternatively, a flow limitation $q_{\text{bound},\mu}(k)$ can be defined, and the virtual downstream density is calculated according to

$$\rho_\mu(k+1) = \begin{cases} \rho_{\text{upstream},n}(k) \ , & \text{if } q_\mu < q_{\text{bound},\mu}(k) \text{ and } \rho_{\text{upstream},n}(k) < \rho_{\text{crit},\mu} \\ \rho_\mu(k) + C_\mu\big(q_\mu(k) - q_{\text{bound},\mu}(k)\big) \ , & \text{else.} \end{cases}$$

where $\rho_{\text{upstream},n}(k)$ is the density of the upstream link, and C_μ is a parameter. If $\rho_\mu(k)$ nor $q_{\text{bound},\mu}(k)$ is defined, then it can be set that

$$\rho_\mu(k) = \rho_{m,N_m}(k) \ .$$

B.4 The destination-dependent mode

For the destination-dependent mode the variable $\gamma_{m,i,j}(k)$ is introduced to express the fraction of traffic on link m, segment i that has destination j. The total density $\rho_{m,i}(k)$ is now decomposed into partial densities $\rho_{m,i,j}(k)$ for each destination j:

$$\rho_{m,i,j}(k) = \gamma_{m,i,j}(k)\rho_{m,i}(k) \ .$$

The conservation equation also becomes destination dependent:

$$\rho_{m,i,j}(k+1) = \rho_{m,i,j}(k) + \frac{T}{L_m\lambda_m}\big(\gamma_{m,i-1,j}(k)q_{m,i-1}(k) - \gamma_{m,i,j}(k)q_{m,i}(k)\big),$$

Origins are also modeled with a destination-dependent queue model. The evolution of the partial queue length $w_{o,j}(k)$ at origin o with destination j is described by:

$$w_{o,j}(k+1) = w_{o,j}(k) + T\big(\gamma_{o,j}(k)d_o(k) - \gamma_{o,j}(k)q_o(k)\big),$$

where $d_o(k)$ is the traffic demand at the origin, $\gamma_{o,j}(k)$ the fraction of the demand traveling to destination j, and $q_o(k)$ the outflow of the origin, and

$$w_o(k) = \sum_{j\in\mathscr{D}_o} w_{o,j}(k),$$

where $\mathscr{D}_o$ is the set of destinations reachable from origin o.

The flow that enters node n is distributed among the leaving links according to

$$Q_{n,j}(k) = \sum_{\mu\in\mathscr{I}_n} q_{\mu,N_\mu}(k)\gamma_{\mu,N_\mu,j}(k)$$

$$q_{m,0}(k) = \sum_{j \in \mathscr{D}_m} Q_{n,j}(k)\beta_{n,m,j}(k),$$
$$\gamma_{m,0,j}(k) = \frac{\beta_{n,m,j}(k)Q_{n,j}(k)}{q_{m,0}(k)},$$

where $Q_{n,j}(k)$ is the total flow that enters the node at simulation step k, N_μ the index of the last segment of link m, $\mathscr{I}_n$ is the set of links that enter node n, $\beta_{n,m,j}(k)$ are the splitting rates (the fraction of the total flow through node n with destination j that leaves via link m), $\mathscr{D}_m$ is the set of destinations reachable from link m, and $q_{m,0}(k)$ is the flow that leaves node n via link m.

Bibliography

[1] *Highway Capacity Manual*, volume 1. Transportation Research Board, 5th edition, 2010.

[2] G. Ambrosino, M. Ariola, U. Ciniglio, F. Corraro, A. Pironti, and M. Virgilio. Algorithms for 3D UAV path generation and tracking. In *Proceeding of the 45th IEEE Conference on Decision and Control (CDC 2006)*, pages 5275–5280, 2006.

[3] A. Atamtürk and M. W. P. Savelsbergh. Integer-programming software systems. *Annals of Operations Research*, 140(1):67–124, 2005.

[4] P. Athol. Interdependence of certain operational characteristics within a moving traffic stream. *Highway Research Record*, 72:58–87, 1965.

[5] T. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary programming, Genetic Algorithms*. Oxford University Press, 1996.

[6] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

[7] T. Bellemans, B. De Schutter, and B. De Moor. Model predictive control for ramp metering of motorway traffic: A case study. *Control Engineering Practice*, 14(7):757–767, 2006.

[8] R. E. Bellman. A Markovian decision process. Technical report, RAND Corporation, 1957.

[9] R. E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[10] E. Beltrami and L. D. Bodin. Networks and vehicle routing for municipal waste collection. *Networks*, 4(1):65–94, 1974.

[11] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[12] D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[13] J. C. Bezdek and R. J. Hathaway. Some notes on alternating optimization. In R. P. Nikhil and S. Michio, editors, *Advances in Soft Computing—AFSS 2002*, pages 288–300. Springer, Calcutta, India, 2002.

[14] R. Bishop. *Intelligent Vehicle Technology and Trends*. Artech House, 2005.

[15] M. C. J. Bliemer and D. H. van Amelsfort. Rewarding instead of charging road users: a model case study investigating effects on traffic conditions. *European Transport\Trasporti Europei*, (44):23–40, 2010.

[16] C. Blum. Ant Colony Optimization: Introduction and recent trends. *Physics of Life Reviews*, 2(4):353–373, 2005.

[17] L. D. Bodin and S. J. Kursh. A computer-assisted system for the routing and scheduling of street sweepers. *Operations Research*, 26(4):525–537, 1978.

[18] E. Bogers, F. Viti, and S. P. Hoogendoorn. Joint modeling of advanced travel information service, habit, and learning impacts on route choice by laboratory simulator experiments. *Transportation Research Record*, (1926):189–197, 2005.

[19] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta Numerica*, 4(1): 1–51, 1995.

[20] A. S. Bortoff. Path planning for UAVs. In *Proceedings of the 2000 American Control Conference (ACC 2000)*, pages 364–368, Chicago, US, June 2000.

[21] J. Bottom, M. Ben-Akiva, M. Bierlaire, I. Chabini, H. N. Koutsopoulos, and Q. Yang. Investigation of route guidance generation issues by simulation with DynaMIT. In A. Ceder, editor, *Transportation and Traffic Theory. Proceedings of the 14th International Symposium on Transportation and Traffic Theory*, pages 577–600. Pergamon, 1999.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[23] D. Braess, A. Nagurney, and T. Wakolbinger. On a paradox of traffic planning. *Transportation Science*, 39(4):446–450, 2005.

[24] B. Bullnheimer, R. F. Hartl, and C. Strauss. A new rank-based version of the Ant System: A computational study. *Central European Journal for Operations Research and Economics*, 7(1):25–38, 1999.

[25] M. Burger, B. De Schutter, and J. Hellendoorn. Micro-ferry scheduling problem with time windows. In *Proceedings of the 2012 American Control Conference (ACC)*, pages 3998–4003, Montréal, Canada, 2012.

[26] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, 2010.

[27] E. F. Camacho and C. B. Alba. *Model Predictive Control*. Springer, 2013.

[28] W. Candler and R. Townsley. A linear two-level programming problem. *Computers and Operations Research*, 9(1):59–76, 1982.

[29] G. E. Cantarella, G. Pavone, and A. Vitetta. Heuristics for urban road network design: Lane layout and signal settings. *European Journal of Operational Research*, 175(3): 1682–1695, 2006.

[30] G. Di Caro and M. Dorigo. AntNet: distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9(1):317–365, 1998.

[31] H. Chen, K. C. Chang, and C. S. Agate. A dynamic path planning algorithm for UAV tracking. *Proceeding of SPIE*, 7336:73360B1–73360B10, 2009.

[32] M. Chen and A. S. Alfa. A network design algorithm using a stochastic incremental traffic assignment approach. *Transportation Science*, 25(3):215–224, 1991.

[33] S. Chiou. Bilevel programming for the continuous transport network design problem. *Transportation Research Part B: Methodological*, 39(4):361–383, 2005.

[34] B. Coifman, M. McCord, R. G. Mishalani, M. Iswalt, and Y. Ji. Roadway traffic monitoring from an unmanned aerial vehicle. In *Proceedings of the IEE Intelligent Transport Systems*, volume 153, pages 11–20, 2006.

[35] Z. Cong, B. De Schutter, and R. Babuška. Ant colony routing algorithm for freeway networks. *Transportation Research Part C*, 37:1–19, December 2013.

[36] Z. Cong, B. De Schutter, and R. Babuška. On the convergence of ant colony optimization with stench pheromone. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pages 1876–1883, Cancún, Mexico, June 2013.

[37] Z. Cong, B. De Schutter, and R. Babuška. Co-design of traffic network topology and control measures, 2014. Accepted with minor modification by Transportation Research Part C.

[38] C. F. Daganzo. *Fundamentals of Transportation and Traffic Operations*. Pergamon, 1997.

[39] G. B. Dantzig and M. N. Thapa. *Linear Programming 1: Introduction*. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, 1997.

[40] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavior*, 3(2):159–168, 1990.

[41] A. C. DeSerpa. A theory of the economics of time. *The Economic Journal*, 81(324): 828–846, 1971.

[42] P. Doherty, G. Granlund, K. Kuchcinski, E. Sandewall, K. Nordberg, E. Skarman, and J. Wiklund. The WITAS unmanned aerial vehicle project. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, pages 747–755, 2000.

[43] M. Dorigo. *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano, Milan, Italy, 1992.

[44] M. Dorigo and L. M. Gambardella. Ant Colony System: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, April 1997.

[45] M. Dorigo and L. M. Gambardella. Ant colonies for the travelling salesman problem. *Biosystems*, 43(2):73–81, 1997.

[46] M. Dorigo and T. Stützle. *Ant Colony Optimization*. Bradford Company, 2004.

[47] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 26(1):1–13, 1996.

[48] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36(3):307–339, 1986.

[49] A. Dussutour, V. Fourcassié, and D. Helbing. Optimal traffic organization in ants under crowded conditions. *Nature*, 428:70–73, March 2004.

[50] A.E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, July 1999.

[51] H. A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems, Part I: The Chinese postman problem. *Operations Research*, 43(2):231–242, 1995.

[52] H. A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems, Part II: The rural postman problem. *Operations Research*, 43(3):399–414, 1995.

[53] D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

[54] C. De Fabritiis, R. Ragona, and G. Valenti. Traffic estimation and prediction based on real time floating car data. In *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems (ITSC 2008)*, pages 197–203, Beijing, China, 2008.

[55] Federal Highway Administration, U.S. Department of Transportation. Freeway Management and Operations Handbook, September 2003.

[56] V. Fourcassié, A. Dussutour, and J. Deneubourg. Ant traffic rules. *The Journal of Experimental Biology*, 213:2357–2363, 2010.

[57] E. Frew, T. McGee, Z. Kim, X. Xiao, J. Jackson, M. Morimoto, S. Rathinam, J. Padial, and R. Sengupta. Vision-based road-following using a small autonomous aircraft. In *Proceedings of the 2004 IEEE Aerospace Conference*, pages 3006–3015, March 2004.

[58] T. L. Friesz, R. L. Tobin, H. J. Cho, and N. J. Mehta. Sensitivity analysis based heuristic algorithms for mathematical programs with variational inequality constraints. *Mathematical Programming*, 48(1-3):265–284, 1990.

[59] L. M. Gambardella, É. Taillard, and G. Agazzi. MACS-VRPTW: A multiple colony system for vehicle routing problems with time windows. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 63–76. McGraw-Hill, London, UK, 1999.

[60] J. Gancet, G. Hattenberger, R. Alami, and S. Lacroix. Task planning and control for a multi-uav system: architecture and algorithms. In *Proceedings of the 2005 IEEE International Conference on Intelligent Robots and Systems (IROS 2005)*, pages 1017–1022, Aug 2005.

[61] Z. Gao, J. Wu, and H. Sun. Solution algorithm for the bi-level discrete network design problem. *Transportation Research Part B: Methodological*, 39(6):479–495, 2005.

[62] M. Gentili and P. B. Mirchandani. Locating sensors on traffic networks: Models, challenges and research opportunities. *Transportation Research Part C: Emerging Technologies*, 24:227–255, 2012.

[63] A. M. Geoffrion. Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237–260, 1972.

[64] P. Geurts, 2010. http://www.montefiore.ulg.ac.be/~geurts/Software.html.

[65] Google. https://www.google.com/selfdrivingcar/.

[66] W. J. Gutjahr. A graph-based ant system and its convergence. *Future Generation Computer Systems*, 16(8):873–888, 2000.

[67] W. J. Gutjahr. ACO algorithms with guaranteed convergence to the optimal solution. *Information Processing Letters*, 82(3):145–153, 2002.

[68] E. Hadjiconstantinou and N. Christofides. An efficient implementation of an algorithm for finding K shortest simple paths. *Networks*, 34(2):88–101, 1999.

[69] W. W. Hardgrave and G. L. Nemhauser. On the relation between the traveling-salesman and the longest-path problems. *Operations Research*, 10(5):647–657, 1962.

[70] A. Hegyi. *Model Predictive Control for Integrating Traffic Control Measures*. PhD thesis, Delft University of Technology, Delft, The Netherlands, February 2004.

[71] A. Hegyi, B. De Schutter, and H. Hellendoorn. Model predictive control for optimal coordination of ramp metering and variable speed limits. *Transportation Research Part C*, 13(3):185–209, 2005.

[72] 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.

[73] R. Hooke and T. A. Jeeves. "Direct search" solution of numerical and statistical problems. *Journal of the ACM*, 8(2):212–229, 1961.

[74] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, pages 19–26, Anaheim, USA, 1993.

[75] T. Hu and H. S. Mahmassani. Day-to-day evolution of network flows under real-time information and reactive signal control. *Transportation Research Part C: Emerging Technologies*, 5(1):51–69, 1997.

[76] O. Huibregtse, S. Hoogendoorn, A. Hegyi, and M. Bliemer. A method to optimize evacuation instructions. *OR Spectrum*, 33(3):595–627, 2011.

[77] T. D. Jorand. Sky count of traffic congestion and demand. *Traffic Engeering and Control*, 7(5):312–315, 1965.

[78] P. Kachroo and K. Özbay. *Feedback Control theory for Dynamic Traffic Assignment*. Springer, 1999.

[79] N. Katoh, T. Ibaraki, and H. Mine. An efficient algorithm for K shortest simple paths. *Networks*, 12(4):411–427, 1982.

[80] H. Kawamura, M. Yamamoto, K. Suzuki, and A. Ohuchi. Multiple ant colonies algorithm based on colony level interactions. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 83(2):371–379, 2000.

[81] J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial & Applied Mathematics*, 8(4):703–712, 1960.

[82] B. S. Kerner, C. Demir, R. G. Herrtwich, S. L. Klenov, H. Rehborn, M. Aleksic, and A. Haug. Traffic state detection with floating car data in road networks. In *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems (ITSC 2005)*, pages 44–49, 2005.

[83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[84] A. Kotsialos and M. Papageorgiou. Efficiency and equity properties of freeway network-wide ramp metering with AMOC. *Transportation Research Part C: Emerging Technologies*, 12(6):401–420, 2004.

[85] A. Kotsialos, M. Papageorgiou, and A. Messmer. Optimal coordinated and integrated motorway network traffic control. In *Proceedings of the 14th International Symposium on Transportation and Traffic Theory*, pages 621–644, Jerusalem, Israel, July 1999.

[86] A. Kotsialos, M. Papageorgiou, C. Diakaki, Y. Pavlis, and F. Middelham. Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool METANET. *IEEE Transactions on Intelligent Transportation Systems*, 3(4):282–292, 2002.

[87] 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.

[88] M. Kwan. Graphic programming using odd or even points. *Chinese Math*, 1(273-277): 110, 1962.

[89] E. L. Lawler and D. E. Wood. Branch-and-bound methods: A survey. *Operations Research*, 14(4):699–719, 1966.

[90] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley New York, 1985.

[91] L. Levy and L. Bodin. The arc oriented location routing problem. *INFOR*, 27(1):74–94, 1989.

[92] M. J. Lighthill and G. B. Whitham. On kinematic waves II: A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 229(1178):317–345, 1955.

[93] J. T. Linderoth and T. K. Ralphs. Noncommercial software for mixed-integer linear programming. In J. K. Karlof, editor, *Integer Programming: Theory and Practice*, pages 253–303. CRC Press, 2005.

[94] F.G. Lobo, C.F. Lima, and Z. Michalewicz. *Parameter Setting in Evolutionary Algorithms*. Springer, 2007.

[95] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 2002.

[96] T. L. Magnanti and R. T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55, 1984.

[97] H.S. Mahmassani, N.N. Huynh, K. Srinivasan, and M. Kraan. Tripmaker choice behavior for shopping trips under real-time information: model formulation and results of stated-preference internet-based interactive experiments. *Journal of Retailing and Consumer Services*, 10(6):311–321, 2003.

[98] A. Messmer and M. Papageorgiou. METANET: A macroscopic simulation program for motorway networks. *Traffic Engineering and Control*, 31(8/9):466–470, 1990.

[99] F. Middelham. State of practice in dynamic traffic management in the Netherlands. In *Proceedings of the 10th IFAC Symposium on Control in Transportation Systems (CTS 2003)*, Tokyo, Japan, August 2003.

[100] P. Miliotis, G. Laporte, and Y. Nobert. Computational comparison of two methods for finding the shortest complete cycle or circuit in a graph. *RAIRO-Operations Research-Recherche Opérationnelle*, 15(3):233–239, 1981.

[101] C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

[102] J. Montgomery and M. Randall. Anti-pheromone as a tool for better exploration of search space. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Ant Algorithms*, pages 100–110. Springer, 2002.

[103] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, 1994.

[104] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, and R. Siegwart. A UAV system for inspection of industrial facilities. In *Proceedings of the 2013 IEEE Aerospace Conference*, pages 1–8, March 2013.

[105] A. Nowé, K. Verbeeck, and P. Vrancx. Multi-type ant colony: The edge disjoint paths problem. In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, editors, *Ant Colony Optimization and Swarm Intelligence*, pages 202–213. Springer, 2004.

[106] M. Papageorgiou. Dynamic modeling, assignment, and route guidance in traffic networks. *Transportation Research Part B*, 24B(6):471–495, 1990.

[107] M. Papageorgiou and A. Kotsialos. Freeway ramp metering: An overview. *IEEE Transactions on Intelligent Transportation Systems*, 3(4):271–280, 2002.

[108] M. Papageorgiou, H. Hadj-Salem, and J. Blosseville. ALINEA: A local feedback control law for on-ramp metering. *Transportation Research Record*, (1320):58–64, 1991.

[109] M. Papageorgiou, J. M. Blosseville, and H. Hadj-Salem. La fluidification des rocades de i'Ile de France: Un projet d'importance. Internal report No. 1998-17, Dynamic Systems and Simulation Laboratory, Technical University of Crete, Chania, Greece, 1998.

[110] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, 2003.

[111] M. Papageorgiou, E. Kosmatopoulos, and I. Papamichail. Effects of variable speed limits on motorway traffic flow. *Transportation Research Record*, 2047:37–48, 2008.

[112] P. M. Pardalos and M. G. C. Resende. *Handbook of Applied Optimization*. Oxford University Press, 2002.

[113] 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.

[114] 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.

[115] H. Poorzahedy and M. A. Turnquist. Approximate algorithms for the discrete network design problem. *Transportation Research Part B: Methodological*, 16(1):45–55, 1982.

[116] W.B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, 2007.

[117] A. Puri. A survey of unmanned aerial vehicles (UAV) for traffic surveillance. Technical report, Department of Computer Science and Engineering, University of South Florida, 2005.

[118] J. B. Rawlings and D. Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2009.

[119] P. I. Richards. Shockwaves on the highway. *Operations Research*, 4(1):42–51, 1956.

[120] M. Riedmiller. Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In *Machine Learning: ECML 2005*, pages 317–328. Springer, 2005.

[121] Y. Sheffi. *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall, 1985.

[122] Y. C. Shin and Y. S. Joo. Optimization of machining conditions with practical constraints. *The International Journal of Production Research*, 30(12):2907–2919, 1992.

[123] K. A. Small and J. A. Gómez-Ibanez. Road pricing for congestion management: The transition from theory to policy. In K. J. Button and E. T. Verhoef, editors, *Road Pricing, Traffic Congestion and the Environment: Issues of Efficiency and Social Feasibility*, pages 213–246. Edward Elgar, Cheltenham, UK, 1998.

[124] S. Smulders. Control of freeway traffic flow by variable speed signs. *Transportation Research Part B: Methodological*, 24(2):111–132, 1990.

[125] S. Srinivasan, H. Latchman, J. Shea, T. Wong, and J. McNair. Airborne traffic surveillance systems: video surveillance of highway traffic. In *Proceedings of the ACM 2nd International Workshop on Video Surveillance & Sensor Networks*, pages 131–135, 2004.

[126] T. Stützle. An ant approach to the flow shop problem. In *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing*, pages 1560–1564, Aachen, Germany, 1998.

[127] T. Stützle and M. Dorigo. A short convergence proof for a class of Ant Colony Optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 6(4):358–365, August 2002.

[128] T. Stützle and H.H. Hoos. MAX-MIN Ant System. *Future Generation Computer Systems*, 16(8):889–914, 2000.

[129] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[130] J. W. Suurballe. Disjoint paths in a network. *Networks*, 4(2):125–145, 1974.

[131] H. A. Taha. *Operations Research: An Introduction, 4th Ed.* Macmillan Publishing Company, 1987.

[132] Z. Tang and U. Ozguner. Motion planning for multitarget surveillance with mobile sensor agents. *IEEE Transactions on Robotics*, 21(5):898–908, 2005.

[133] M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2):225–249, 2005.

[134] P. Theodorakopoulos and S. Lacroix. A strategy for tracking a ground target with a UAV. In *Proceedings of the 2008 International Conference on Intelligent Robots and Systems (IROS 2008)*, pages 1254–1259, 2008.

[135] R. L. Tobin and T. L. Friesz. Sensitivity analysis for equilibrium network flow. *Transportation Science*, 22(4):242–250, 1988.

[136] C. O. Tong and S. C. Wong. A predictive dynamic traffic assignment model in congested capacity-constrained road networks. *Transportation Research Part B: Methodological*, 34(8):625–644, 2000.

[137] U.S. General Accounting Office. Comparison of states' highway construction costs. Technical report, November 2003.

[138] Y. Wang, M. Papageorgiou, and A. Messmer. A predictive feedback routing control strategy for freeway network traffic. In *Proceedings of the American Control Conference (ACC 2002)*, pages 3606–3611, Anchorage, Alaska, May 2002.

[139] M. Wardman. The value of travel time: a review of British evidence. *Journal of Transport Economics and Policy*, 32(3):285–316, 1998.

[140] J. G. Wardrop. Some theoretical aspects of road traffic research. *ICE Proceedings: Engineering Divisions*, 1(3):325–362, 1952.

[141] B. Williams. Highway control. *IEE Review*, 42(5):191–194, 1996.

[142] S. J. Wright. *Primal-Dual Interior Point Methods*. SIAM, 1997.

[143] G. Yang and V. Kapila. Optimal path planning for unmanned air vehicles with kinematic and tactical constraints. In *Proceedings of the 41st IEEE Conference on Decision and Control (CDC 2002)*, pages 1301–1306, Las Vegas, December 2002.

[144] H. Yang. Sensitivity analysis for the elastic-demand network equilibrium problem with applications. *Transportation Research Part B: Methodological*, 31(1):55–70, 1997.

[145] H. Yang and M. G. H. Bell. Models and algorithms for road network design: a review and some new developments. *Transport Reviews*, 18(3):257–278, 1998.

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

[147] 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.

[148] W. Zhu, K. Boriboonsomsin, and M. Barth. Defining a freeway mobility index for roadway navigation. *Journal of Intelligent Transportation Systems*, 14(1):37–50, 2010.

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.

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, Özgül, *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

Schakel, W., *Development, Simulation and Evaluation of In-car Advice on Headway, Speed and Lane*, T2015/9, May 2015, TRAIL Thesis Series, the Netherlands

Dorsser, J.C.M. van, *Very Long Term Development of the Dutch Inland Waterway Transport System: Policy analysis, transport projections, shipping scenarios, and a new perspective on economic growth and future discounting*, T2015/8, May 2015, TRAIL Thesis Series, the Netherlands

Hajiahmadi, M., *Optimal and Robust Switching Control Strategies: Theory, and applications in traffic management*, T2015/7, April 2015, TRAIL Thesis Series, the Netherlands

Wang, Y., *On-line Distributed Prediction and Control for a Large-scale Traffic Network*, T2015/6, March 2015, TRAIL Thesis Series, the Netherlands

Vreeswijk, J.D., *The Dynamics of User Perception, Decision Making and Route Choice, T2015/5,*

February 2015, TRAIL Thesis Series, the Netherlands Lu, R., The Effects of Information and Communication Technologies on Accessibility, T2015/4, February 2015, TRAIL Thesis Series, the Netherlands

Ramos, G. de, *Dynamic Route Choice Modelling of the Effects of Travel Information using RP Data*, T2015/3, February 2015, TRAIL Thesis Series, the Netherlands

Sierzchula, W.S., *Development and Early Adoption of Electric Vehicles: Understanding the tempest*, T2015/2, January 2015, TRAIL Thesis Series, the Netherlands

Vianen, T. van, *Simulation-integrated Design of Dry Bulk Terminals*, T2015/1, January 2015, TRAIL Thesis Series, the Netherlands

Risto, M., *Cooperative In-Vehicle Advice: A study into drivers' ability and willingness to follow tactical driver advice*, T2014/10, December 2014, TRAIL Thesis Series, the Netherlands

Djukic, T., *Dynamic OD Demand Estimation and Prediction for Dynamic Traffic Management*, T2014/9, November 2014, TRAIL Thesis Series, the Netherlands

Chen, C., *Task Complexity and Time Pressure: Impacts on activity-travel choices*, T2014/8, November 2014, TRAIL Thesis Series, the Netherlands

Wang, Y., *Optimal Trajectory Planning and Train Scheduling for Railway Systems*, T2014/7, November 2014, TRAIL Thesis Series, the Netherlands

Wang, M., *Generic Model Predictive Control Framework for Advanced Driver Assistance Systems*, T2014/6, October 2014, TRAIL Thesis Series, the Netherlands

Kecman, P., *Models for Predictive Railway Traffic Management*, T2014/5, October 2014, TRAIL Thesis Series, the Netherlands

Davarynejad, M., *Deploying Evolutionary Metaheuristics for Global Optimization*, T2014/4, June 2014, TRAIL Thesis Series, the Netherlands

Li, J., *Characteristics of Chinese Driver Behavior*, T2014/3, June 2014, TRAIL Thesis Series, the Netherlands

Mouter, N., *Cost-Benefit Analysis in Practice: A study of the way Cost-Benefit Analysis is perceived by key actors in the Dutch appraisal practice for spatial-infrastructure projects*, T2014/2, June 2014, TRAIL Thesis Series, the Netherlands

Ohazulike, A., *Road Pricing mechanism: A game theoretic and multi-level approach*, T2014/1, January 2014, TRAIL Thesis Series, the Netherlands

Cranenburgh, S. van, *Vacation Travel Behaviour in a Very Different Future*, T2013/12, November 2013, TRAIL Thesis Series, the Netherlands

Samsura, D.A.A., *Games and the City: Applying game-theoretical approaches to land and property development analysis*, T2013/11, November 2013, TRAIL Thesis Series, the Netherlands

Summary

Efficient Optimization Methods for Freeway Management and Control

Due to the rapid growth of human population, and jobs being distributed unevenly in different locations, daily commuting is required more than ever, which in its turn is creating a huge socio-economic issue: traffic congestion. In order to prevent, or at least to alleviate this problem, traffic management and control is urgently required.

This thesis develops three different management and control methods to improve the performance of traffic networks, with a particular focus on freeway networks, namely

- ant colony optimization for dynamic traffic routing;
- co-design of network topology and control measures;
- path planning of unmanned aerial vehicles for monitoring traffic networks.

Usually, solving these problems for a large-scale freeway network will result in an extremely high computational burden. The main contribution of this thesis consists in the development of solution methods for these problems to solve them efficiently with a well-balanced trade-off between performance and computation speed. Each method is summarized as follows:

- **Ant colony optimization for dynamic traffic routing**
  We propose a dynamic traffic routing algorithm, called Ant Colony Routing (ACR). The algorithm is developed based on Ant Colony Optimization, which is an optimization algorithm that mimics the behavior of ants seeking the shortest path between their nest and a source of food. The ACR algorithm contains two parts: network pruning and model predictive control. The network pruning step is used to remove some "unnecessary" links and nodes from the original freeway network, in order to increase the computation speed. The model predictive control step involves using artificial ants to determine traffic flows in the reduced network. More specially, at each control step, we first map the reduced network into an ant graph, then use a concept called stench pheromone to disperse ants to different paths in the ant graph, and when ants finish searching the graph, the resulting assignment of ants in the ant graph is used to determine the splitting rates for flows in the freeway network. The ACR algorithm can effectively guide the vehicles from multiple origins to multiple destinations in a traffic network.

- **Co-design of network topology and control measures**
  We consider a co-design method that jointly optimizes the topology and the control

measures in a freeway network. Usually, co-design involves a nonlinear, non-convex optimization problem with mixed-integer variables, which will be computationally expensive when dealing with a large-scale network and for multiple control measures. Therefore, we discuss four different solution frameworks that can be used for solving this problem, according to different requirements on the computational complexity and speed, namely separate optimization, iterative optimization, bi-level optimization, and joint optimization. The results show that the separate optimization is the most computationally efficient, but the other three approaches have much better performance.

- **Path planning of unmanned aerial vehicles for monitoring networks**
  We address a path planning problem involving unmanned aerial vehicles (UAVs) for monitoring freeway networks. We consider two different monitoring settings: UAVs can monitor while flying, and UAVs can only monitor when hovering. In the first setting, the monitoring problem is formulated as a periodical multiple rural postman problem, and solved using mixed-integer linear programming. In the second setting, the problem is formulated as a Markov decision process, and solved by three different solution methods, namely the fitted Q-iteration, the model predictive control, and the parameterized control. Both methods can nearly achieve a full coverage of the network.

Zhe Cong

Samenvatting

Efficiënte optimalisatiemethoden voor het managen en regelen van autowegen

Door de snelle groei van de menselijke bevolking en banen die ongelijk verdeeld zijn over verschillende locaties, is dagelijks pendelen meer dan ooit noodzakelijk, wat op zijn beurt een groot sociaaleconomisch probleem creëert: verkeersopstoppingen. Om dit probleem te voorkomen, of in ieder geval te verzachten, is management en regeling van snelwegen dringend nodig.

Dit proefschrift ontwikkelt drie verschillende management en regelmethoden om de prestaties van verkeersnetwerken te verbeteren, met een specifieke focus op netwerken van autowegen, namelijk

- mierenkolonie-optimalisatie voor dynamische verkeersroutering;
- co-ontwerp van netwerk topologie en regelacties;
- routeplanning van onbemande luchtvaartuigen.

Meestal resulteert het oplossen van deze problemen voor een grootschalig netwerk van autowegen in een extreem hoge rekenkundige belasting. De voornaamste bijdrage van dit proefschrift bestaat uit de ontwikkeling van efficiënt oplossungen van deze problemen met een goed gebalanceerde afweging tussen prestaties en rekensnelheid. De methodes kunnen als volgt kort samengevat worden:

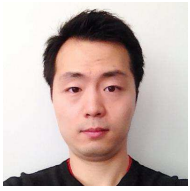
- **mierenkolonie-optimalisatie voor dynamische verkeersroutering**
  We stellen een dynamisch verkeersroutering-algoritme voor, genaamd Ant Colony Routing (ACR). Dit algoritme bestaat uit twee delen: netwerk reductie en model-gebaseerde voorspellende regeling. Het reduceren van het netwerk wordt gebruikt om "overbodig" verbindingen en knooppunten uit het originele autowegen netwerk te verwijderen om zo de rekensnelheid te verhogen. De model-gebaseerde voorspellende regelnstap maakt gebruik van kunstmatige mieren om de verkeersstromen in het gereduceerde netwerk te bepalen. Tijdens elke regelstap vertalen we doorbij eerst het gereduceerde netwerk in een mierengraaf. Daarna gebruiken we een concept genaamd stank-feromoon om de mieren te verspreiden over de verschillende paden in de mierengraaf, en zodra de mieren klaar zijn met het doorzoeken van de graaf wordt de resulterende toewijzing van mieren in de mierengraaf gebruikt om de verdeelratio's van de stromen in het autowegen netwerk te bepalen. Het ACR algoritme kan voertuigen effectief geleiden van verschillende vertrekpunten naar verschillende bestemmingen in een autowegen netwerk.

- **Co-ontwerp van netwerktopologie en regelacties**
  We beschouwen een co-ontwerp-methode dat gezamenlijk de topologie en de regelacties in een autowegen netwerk optimaliseert. Meestal wordt voor co-ontwerp een niet-lineair, niet-convex optimalisatieprobleem met geheeltallige en reële variabelen gebruikt, wat rekenkundig gezien erg zwaar is voor grootschalige netwerken en met een groot aantal regelacties. Om dit probleem aan te pakken gebruiken we vier verschillende oplossingskaders, namelijk aparte optimalisatie, iteratieve optimalisatie, tweelaags optimalisatie, en gezamenlijke optimalisatie. De resultaten laten zien dat aparte optimalisatie rekenkundig het meest efficient is, maar de overigen methoden een veel betere prestatie leveren

- **Route planning van onbemande luchtvaartuigen**
  We pakken het routeplanning-probleem voor de onbemande luchtvaartuigen (in het Engels: Unmanned Aerial Vehicles, UAV's), die gebruikt worden om toezicht te houden over het netwerk van autowegen, aan. We beschouwen twee verschillende situaties voor het toezicht – UAV's kunnen toezicht houden terwijl ze vliegen, en UAV's kunnen alleen toezicht houden als ze stationair zweven. In de eerste situatie wordt het toezichtsprobleem geformuleerd als een periodiek meervoudig landelijk postbode probleem en wordt opgelost door middel van lineaie optimalisatie met geheeltallige en reële variabelen. In de tweede situatie wordt het probleem geformuleerd als een Markov beslissingsproces en wordt het opgelost door middel van drie verschillende oplossingsmethodes, namelijk *fitted Q-iteration*, modelgebaseerde voorspellende regeling, en geparameteriseerde regeling. Beide methodes kunnen een vrijwel volledige dekking van het netwerk bereiken.

Zhe Cong

Curriculum Vitae



Zhe Cong was born in February 1985, Wuhan, Hubei Province, China. In 2007, he obtained the B.Sc degree from the department of automation in Huazhong University of Science and Technology, China. At the same year, he continued his study as a master student in the same department of the same university. After two years, he completed the master program, and received the M.Sc degree of control theory and engineering.

In October 2009, he was sponsored by the Chinese Scholarship Council to become a Ph.D candidate at Delft Center for Systems and Control, Delft University of Technology. In his Ph.D project, he worked on developing efficient optimization algorithms for control measures in large-scale freeway networks, under the supervision of Prof. dr. ir. Bart De Schutter and Prof. dr. Robert Babuška.