A Self-learning Motorway Traffic Control System for Ramp Metering

Chao Lu

Submitted in accordance with the requirements for the degree of
Doctor of Philosophy

The University of Leeds
Institute for Transport Studies

September, 2014

The candidate confirms that the work submitted is his own, except where work which has formed part of jointly-authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Two jointly authored papers are related to this thesis:

Lu, C., Chen, H. and Grant-Muller, S. (2013) An indirect reinforcement learning approach for ramp control under incident-induced congestion. *16th International IEEE Conference on Intelligent Transportation Systems, October 2013, The Hague, Netherlands*. pp. 979-984.

Lu, C., Chen, H. and Grant-Muller, S. (2014) Indirect reinforcement learning for incident-responsive ramp control. *Procedia-Social and Behavioral Sciences, 111,* pp. 1112-1122.

These two papers contain partial information from Chapters 2, 3, 4 and full information from Chapter 8 of the thesis. The major contributions to these papers regarding the idea and writing were made by the author of this thesis. Two co-authors gave suggestions about the paper organisation and polishing, and they are both PhD supervisors of the candidate.

ACKNOWLEDGEMENTS

Firstly, I would like to express my gratitude to my supervisors, Dr. Haibo Chen and Dr. Susan Grant-Muller, for their patient guidance, support and help during my PhD journey.

Secondly, I would like to thank the China Scholarship Council (CSC) and University of Leeds for funding this PhD project.

Here, many thanks to Dr. James Tate who shared the AIMSUN software with me. Then, I would like to thank Said Munir, Andrew Tomlinson and Mojtaba Moharra for suggestions about time management in the final stage of PhD. I would also like to thank Padma Seetharaman, Afzal Ahmed, Izza Anwer, Chris Leahy and Ian Philips for sharing useful information with me at different stages of my research.

Very special thanks should go to Jie Huang who always encourages me and helps me get through difficult times. I would like to express my deepest gratitude to all my family members, especially my parents, who are always standing behind me, supporting me and giving me endless love during my life. Without their constant support and encouragement, this thesis would not have been accomplished.

Finally, I am thankful to everyone who has given me support and help since I started my PhD study, which makes my PhD journey enjoyable and worth treasuring forever.

ABSTRACT

Self-learning systems have attracted increasing attention in the ramp metering domain in recent years. These systems are based on reinforcement learning (RL) and can learn to control motorway traffic adaptively. However, RL-based ramp metering systems are still in their early stages and have shown limitations regarding their design and evaluation. This research aims to develop a new RL-based system (known as RAS) for ramp metering to overcome these limitations.

A general framework for designing a RL-based system is proposed in this research. It contains the definition of three RL elements in a ramp metering scenario and a system structure which brings together all modules to accomplish the reinforcement learning process. Under this framework, two control algorithms for both single- and multi-objective problems are developed. In addition, to evaluate the proposed system, a software platform combining the new system and a traffic flow model is developed in the research. Based on the platform developed, a systematic evaluation is carried out through a series of simulation-based experiments.

By comparing with a widely used control strategy, ALINEA, the proposed system, RAS, has shown its effectiveness in learning the optimal control actions for different control objectives in both hypothetical and real motorway networks. It is found that RAS outperforms ALINEA on improving traffic efficiency in the situation with severe congestion and on maintaining user equity when multiple on-ramps are included in the motorway network. Moreover, this research has been extended to use indirect learning technology to deal with incident-induced congestion. Tests for this extension to the work are carried out based on the platform developed and a commercial software package, AIMSUN, which have shown the potential of the extended system in tackling incident-induced congestion.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF NOTATIONS

A	Action set
A_I	Action set of agent I
a	Action variable for action set
a_{greedy}^t	Greedy action at control step t
a_I^t	Greedy action for agent I at control step t
$a_{I,greedy}^t$	Action variable for action set A_I at control step t
C_I	Set of discrete metering rates for agent I
c_i^k	Calculated metering rate for cell i at time step k
$c_i^{\max}$	The maximum metering rate
$c_i^{\min}$	The minimum metering rate
D_i^k	Average delay of users from on-ramp i at step time k
d_i	Delay of user i
d_{on}	On-ramp demand
d_{off}^n	Off-ramp departure (exit) flow in the non-controlled and controlled situations
d_{off}^c	Off-ramp departure (exit) flow in controlled situation
d_{on}^k	On-ramp demand at time step k
$d_{on,i}^k$	On-ramp demand of cell i at time step k
$d_{off,i}^k$	Off-ramp departure (exit) flow of cell i at time step k
$d_{off,i}^{\max}$	The maximum off-ramp departure (exit) flow of cell i
$d_{on,i}^{up}$, $d_{on,i}^{low}$	Upper and lower bounds of $d_{on,i}^k$ (used for state mapping)
$\Delta d_{on,i}$	State interval for $d_{on,i}^k$ (used for state mapping)
K_R	Regulatory parameter ($K_R > 0$)
l_i	Length of cell i

m_r	On-ramp flow
$m_r^{\min}$	The minimum on-ramp flow
m_r^k	On-ramp flow at time step k
$m_{r,i}^k$	On-ramp flow that enters the mainline of cell i at step k
$m_r'^k$	Metering rate calculated by queue management algorithm at time step k
m_{fr}^k	Final determined metering rate, $m_{fr}^k = \max\{m_r^k, m_r'^k\}$
N_I	Number of states in state set S_I
$N_{nmain,I}$	Number of states in state set $S_{nmain,I}$
$N_{qin,I}$	Number of states in state set $S_{qin,I}$
$N_{non,I}$	Number of states in state set $S_{non,I}$
$N_{don,I}$	Number of states in state set $S_{don,I}$
$N_{A,I}$	Number of actions in the action set A_I
N_{cs}	Number of simulation steps within one control interval
N_j	Number of objectives
N_e	Number of episodes
N_t	Number of control steps
N_{on}	Number of on-ramps
n_{main}^k	Number of vehicles on the mainline at time step k
n_{on}^k	Number of vehicles on the on-ramp (on-ramp queue length) at time step k
n_i^k	Number of vahicles of cell i at time step k
$n_{main,i}^k$	Number of vehicles on the mainline of cell i at time step k
$n_{on,i}^k$	Number of vehicles on the on-ramp (on-ramp queue length) of cell i at time step k
$n_{main,i}^{\max}$	The maximum number of vehicles that can stay on the mainline of cell i
$n_{on,i}^{\max}$	The maximum number of vehicles that can stay on the on-ramp of cell i

$n_{main,i}^{up}$, $n_{main,i}^{low}$	Upper and lower bounds of $n_{main,i}^{k}$ (used for state mapping)
$n_{on,i}^{up}$, $n_{on,i}^{low}$	Upper and lower bounds of $n_{on,i}^{k}$ (used for state mapping)
$\Delta n_{main,i}$, $\Delta n_{on,i}$	State intervals for $n_{main,i}^{k}$ and $n_{on,i}^{k}$ (used for state mapping)
n'_{on}	On-ramp queue constraint
$\hat{n}_{main}^{k}$	Queue increase on the mainline at time step k
$\hat{n}_{on}^{k}$	Queue increase on the mainline at time step k
$\hat{n}_{main}^{\max}$	The maximum queue increase on the mainline
$\hat{n}_{main}^{\min}$	The minimum queue increase on the mainline
$\hat{n}_{on}^{\max}$	The maximum queue increase on the on-ramp
$\hat{n}_{on}^{\min}$	The minimum queue increase on the on-ramp
o_{crit}	Critical occupancy of downstream motorway
$\hat{o}$	Target downstream occupancy
o_{out}^{k}	Occupancy measured downstream at time step k
$P(s' \mid s,a)$	State transition probability, representing the probability of reaching state s' after executing action a at state s
$p(a \mid s)$	Action selection probability for action a at state s
$Q^{\pi}(s,a)$	Action-value function (Q value) for policy π and state-action pair (s,a)
$Q^{t}(s,a)$	Q value for state-action pair (s,a) updated at control step t
$Q_{j}^{t}(s,a)$	Q value for objective j with state-action pair (s,a) updated at control step t
$Q_{I,j}^{t}(s,a)$	Q value for agent I with state-action pair (s,a) updated at control step t for objective j
q	Flow
q_{cap}	Road capacity
q_{dis}	Queue discharge rate
q_{inc}	Incident capacity
$q_{S,i}{}^{k}$	Number of vehicles that can be sent by cell i duting T_s
$q_{R,i}{}^{k}$	Number of vehicles that can be received by cell i duting T_s
$q_{ij}{}^{k}$	Flow coming from cell i and entering cell j

$q_{in,i}^{k}$	Mainline inflow of cell i at time step k
$q_{out,i}^{k}$	Mainline outflow of cell i at time step k
$q_{cap,i}$	Mainline capacity of cell i at time step k
$q_{dis,i}$	Queue discharge rate of cell i
$q_{in,i}^{up}$, $q_{in,i}^{low}$	Upper and lower bounds of $q_{in,i}^{k}$ (used for state mapping)
$\Delta q_{in,i}$	State interval for $q_{in,i}^{k}$ (used for state mapping)
$R(s,a,s')$	Reward function, corresponding to an immediate reward for the agent reaching state s' from state s after taking action a
$R(s,a)$	Expected reward for the agent at state s after taking action a
Re^{t}	Return at control step t
r	Immediate reward , $r \in \mathbb{R}$
$r_{I,j}^{t}$	Immediate reward received by agent I at control step t for objective j
$r_{raw,I,j}^{t}$	Raw reward value received by agent I at control step t for objective j
$r_{raw,I,j}^{\max}$	The maximum raw reward value received by agent I at control step t for objective j
$r_{raw,I,j}^{\min}$	The minimum raw reward value received by agent I at control step t for objective j
S	State set
S_I	State set (or state space) for agent I
$S_{nmain,I}$	State set for the number of vehicles on the mainline of cell I corresponding to agent I
$S_{qin,I}$	State set for the mainline inflow of cell I corresponding to agent I
$S_{non,I}$	State set for the number of vehicles on the on-ramp of cell I corresponding to agent I
$S_{don,I}$	State set for the on-ramp demand flow of cell I corresponding to agent I
s	State variable for state set S
s_I^{t}	State variable for state set S_I at control step t
$s_{nmain,I}^{t}$	State variable for state set $S_{nmain,I}$ at control step t

$s^t_{qin,I}$	State variable for state set $S_{qin,I}$ at control step t
$s^t_{non,I}$	State variable for state set $S_{non,I}$ at control step t
$s^t_{don,I}$	State variable for state set $S_{don,I}$ at control step t
$SQ^t(s,a)$	Scalarised Q value for state-action pair (s,a) updated at control step t
$SQ^t_I(s,a)$	Scalarised Q value for agent I with state-action pair (s,a) updated at control step t
T	Time interval between two time steps
T_s	Simulation interval
T_c	Control interval
TTS^n	TTS in the non-controlled situation
TTS^c	TTS in the controlled situation
$V^\pi(s)$.	State-value function for policy π at state s
$V^i(s)$	Value of state s updated in the i th iteration
$V^*(s)$	Optimal value of state s
v	Speed
v_{free}	Free-flow speed
v_{crit}	Critical speed
v_i	Free-flow speed in cell i
w_i	Congestion wave speed in cell i. This speed is the wave propagation speed in the fully congested situation.
x_{ij}	The proportion of vehicles that came from the on-ramp of segment i and passed through segment j
α	Learning rate, $\alpha \in [0,1]$
β	Split ratio, the proportion of mainline flow that exists the motorway from the off-ramp, $\beta \in [0,1)$
β_i	Split ratio of cell i, $\beta_i \in [0,1)$
β_{ij}	Split ratios determining the portion of flow coming from cell i and entering cell j
γ	Discount rate, $\gamma \in [0,1]$
δ_j	Weight value for objective j

ε	Action selection parameter, $\varepsilon \in [0,1]$
η_i	Flow allocation parameter of cell i, $\eta_i \in [0,1]$
θ_i	Flow blending parameter of traffic flow from the on-ramp to the mainline of cell i, $\theta_i \in [0,1]$
λ	Capacity drop parameter (the proportion of road capacity that is still available after capacity drop), $\lambda \in [0,1]$
ξ	Weight, indicating the importance of traffic on the mainline and on-ramp, $\xi \in [0,1]$
π	Policy, a mapping from observed states to executable actions, $\pi : S \to A$
$\pi^*(s)$	Optimal policy at state s
ρ	Density
$\hat{\rho}$	Target density
ρ_{crit}	Critical density
ρ_{jam}	Jam density
ρ_i^k	Density of cell i at time step k
$\rho_{crit,i}$	Critical density of cell i
$\rho_{jam,i}$	Jam density of cell i
$\arg\max$	Argument that makes the given function reach its maximum value
$\max$	The maximum value of a given function
$E\{\}$	Expectation
$\max\{\}$	The maximum value of a set of given numbers
$\min\{\}$	The minimum value of a set of given numbers
$\mathrm{mid}\{\}$	The middle value of three given numbers

CHAPTER 1 INTRODUCTION

The first chapter of this thesis gives a brief overview of the research which is about the design and evaluation of a new self-learning system for ramp metering control. Section 1.1 of this chapter begins with an introduction to the background of ramp metering systems, in particular the recent control systems that have the “self-learning” capability. Then, the limitations of previous studies and research objectives proposed to overcome these limitations are outlined in Sections 1.2 and 1.3. Finally, Section 1.4 gives the organisation of the thesis.

1.1 Background

Traffic congestion has been recognised as one of the main issues affecting daily traffic operation on both urban and inter-urban networks, which occurs when the traffic demand for a road network approaches or exceeds its available road capacity. The cost of traffic congestion in the UK is estimated to range from £2 billion per year (Dodgson et al. 2002) to £20 billion per year according to the Confederation of British Industry (Grant-Muller and Laird 2007). In addition to these economic costs, there are also many other adverse impacts related to traffic congestion, such as reduced safety, increased air pollution and resultant health problems (Han and Naeher 2006, Noland and Quddus 2005). Therefore, managing and controlling traffic congestion has become one of the main concerns of the transport community.

In the inter-urban networks (i.e. motorways), the need for suitable control and management of traffic congestion is even more urgent, as motorways were originally designed to provide high mobility and guarantee orderly traffic operation (Papageorgiou and Kotsialos 2002). To alleviate traffic

congestion and reduce its adverse impacts on motorways, a number of traffic control systems and devices have been developed such as ramp metering, variable speed limits (VSL) and variable message signs (VMS). Among these control measures, ramp metering control has been identified as one of the most effective and efficient methods after more than 50 years application (Zhang and Wang 2013). In the UK, it has been reported that ramp metering can reduce journey times of motorway users by an average of 13% (Highways Agency 2007).

In recent decades, a great number of ramp metering strategies have been proposed to control motorway traffic, from the early fixed-time method mentioned in (Wattleworth and Berry 1965), to traffic responsive strategies such as the capacity-density method (Masher et al. 1975), ALINEA (Asservissement Linéaire d'Entrée Autoroutière) and its variations (Papageorgiou et al. 1991, Papageorgiou and Kotsialos 2002, Smaragdis and Papageorgiou 2003), up to the recent optimisation-based approaches such as AMOC (Advanced Motorway Optimal Control) (Kotsialos et al. 2001), model predictive control methods (Hegyi et al. 2005, Papamichail et al. 2010) and other optimal control methods (Gomes and Horowitz 2006, Zhang and Wang 2013). Among these strategies, the optimisation-based method has become increasingly popular in recent studies, as it is sound and can solve ramp metering problems based on optimisation theory.

Most existing optimisation-based methods are model-based methods which use a traffic flow model to predict traffic conditions and generate optimal control actions based on these predictions to maximise or minimise some predefined control objectives (e.g. maximise motorway throughput or minimise delays) (Hegyi et al. 2005, Papamichail et al. 2010). One limitation of these methods is that they rely on a specific model and have poor adaptability when a mismatch between the used model and the real traffic

condition exists (Davarynejad et al. 2011, Jacob and Abdulhai 2010, Rezaee et al. 2012). In order to overcome this limitation, the “self-learning” concept based on reinforcement learning (RL) was recently proposed by Jacob and Abdulhai (Jacob and Abdulhai 2006, Jacob and Abdulhai 2010) .

RL is a model-free optimisation method, which means it is independent from any traffic flow models (Davarynejad et al. 2011, Jacob and Abdulhai 2010, Rezaee et al. 2012). Given new traffic models or even real road traffic conditions, RL can learn the optimal control actions from them without many adjustments, that is why it is also known as a “self-learning” method. Because of this self-learning capability, the RL-based system can continuously learn to improve itself and adapt to new traffic conditions. In addition to good adaptability, the RL-based system also has high scalability (El-Tantawy et al. 2013, Fares and Gomaa 2015). A memory base is maintained by each RL-based agent (a controller that can control the motorway traffic) and used to record the values of different control actions under different traffic conditions. This memory base can be shared with new agents, as long as they have the same structure. In this way, the RL-based system can be easily extended to involve new agents.

Because of the aforementioned features, the RL-based system has attracted increasing attention in recent years. After the first contribution of Jacob and Abdulhai, some recent studies have also explored the use of RL to solve ramp metering problems under different settings and conditions. For instance, a RL-based system with the ability to manage on-ramp queue was developed in (Davarynejad et al. 2011), local ramp metering control using RL was studied in (Rezaee et al. 2012), coordinated ramp metering with RL was tested in (Veljanovska et al. 2012, Veljanovska et al. 2010) and an incident-responsive RL system for ramp metering was explored in (Lu et al. 2013, Lu et al. 2014).

1.2 Research Problems

Although the aforementioned studies have shown some positive results of using the RL-based system, there remain some limitations in the current applications of RL:

(1) There is a lack of a general framework for designing a RL-based system for ramp metering application, and each study has its own way to define RL elements.

(2) Although a few studies have considered the coordination problems in a RL-based strategy, improving motorway traffic efficiency is still the main concern. How to add new objectives such as user equity and balance different control objectives have not been well studied.

(3) There is a lack of systematic evaluation for a RL-based system regarding the influence of learning parameters and the effectiveness of algorithms on different networks.

This section only gives an outline of these limitations, and the detailed problems related to each limitation will be discussed further in Chapter 2 after a review of the current RL-based systems in the ramp metering domain.

1.3 Research Objectives

To overcome the limitations presented in Section 1.2, a new self-learning system based on RL is developed in this study to deal with ramp metering problems with the following research objectives:

(1) To investigate the state of the art of RL technology and its applications in the ramp metering domain, including both local and coordinated RL-based systems.

(2) To provide a general framework for designing a RL-based ramp metering system, regarding the definitions of RL elements, the structure and modules of a RL-based system.

(3) To explore the application of RL to ramp metering for both single- and multi-objective problems under the framework proposed in Objective (2). Two different control objectives with two control algorithms are developed and analysed.

(4) To develop a platform with initial software implementations based on Objectives (2) and (3), which can be used to evaluate the RL-based system.

(5) To evaluate the proposed system based on Objective (4) by conducting simulation-based experiments considering both hypothetical and real traffic networks.

Objective (1) is the basis of all other objectives mentioned above. Problem (1) introduced in Section 1.2 can be solved by achieving Objective (2), and Objective (3) corresponds to a solution to Problem (2). Besides that, Problem (3) can be tackled by attaining Objectives (4) and (5). Each objective mentioned here will be explained further in Chapter 2 after an analysis of each problem.

1.4 Thesis Organisation

To achieve five research objectives, this thesis is divided into nine chapters presenting the whole design and evaluation process of a self-learning ramp metering system. The organisation of these chapters and their connections can be seen from Figure 1.1.

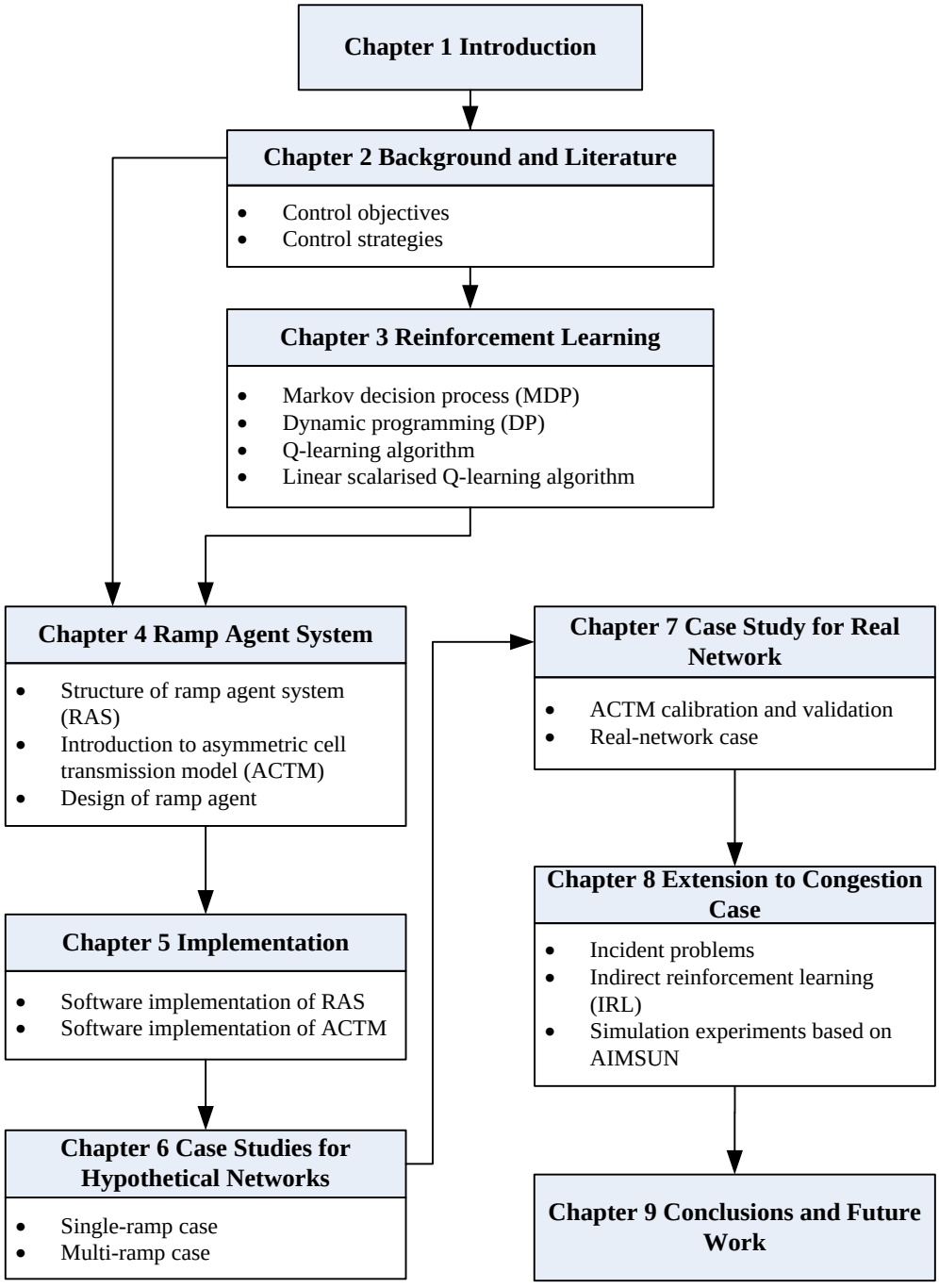


Figure 1.1: Thesis Organisation

Besides the first chapter introduced here, the remainder of this thesis is organised as follows:

Chapter 2 reviews the state of the art in ramp metering domain, including control objectives and related control strategies (especially the RL-based strategies) that can achieve these objectives.

Chapter 3 introduces the reinforcement learning technology in terms of its mechanisms, algorithms and multi-objective learning methods.

Chapter 4 introduces the main work of this research. A general framework for designing a self-learning system is provided in this chapter. This framework contains a general definition of three RL elements, structure and modules used to accomplish the learning process. Besides that, two control algorithms based on the RL mechanism are developed to deal with both single- and multi-objective problems for ramp metering.

Chapter 5 presents the software implementation of the proposed system and a traffic flow model used for evaluation. The C++ implementation of three reusable classes with related sub-classes and functions are introduced in this chapter.

Chapter 6 presents two case studies based on hypothetical networks, i.e. single-ramp and multi-ramp case, to evaluate the proposed system using a macroscopic traffic flow model. Various abilities of the new system such as improving traffic efficiency, managing on-ramp queues and maintaining user equity are tested in this chapter.

Chapter 7 presents the case study based on a real network selected from the M6 motorway in the UK. The ability of the self-learning system to deal with real fluctuating traffic flows is tested in this chapter.

Chapter 8 gives an extension to the basic system developed in Chapter 4 to deal with non-recurrent congestion caused by incidents. Some initial tests for this strategy are carried out using AIMSUN.

Chapter 9 gives conclusions of the whole thesis and discusses possible directions for future work.

CHAPTER 2 BACKGROUND AND LITERATURE

This chapter reviews the main work related to ramp metering in terms of its control objectives and strategies. In Section 2.1, some background knowledge of ramp metering and traffic flow theory is introduced. A discussion of different control objectives of ramp metering is given in Section 2.2. Then, Section 2.3 reviews the state of the art of ramp metering strategies that can achieve these objectives. Finally, Section 2.4 summarises this chapter and further discusses the limitations of RL-based applications.

2.1 Background of Ramp Metering

Before the introduction of detailed control objectives and strategies of ramp metering, some background knowledge including a general introduction of ramp metering and related terminologies in the traffic flow theory is summarised in this section.

2.1.1 Ramp metering problem

Figure 2.1 shows an example of the relationship between ramp metering strategies and a typical motorway segment with one mainline and two linked ramps (i.e. one on-ramp and one off-ramp)[1]. The ramp metering problem mentioned in this study refers to the on-ramp metering control which uses a ramp meter (a signal device) located at the on-ramp to regulate the metering

---

[1] In the UK, the motorway mainline is also named the main carriageway, while the on- and off-ramp are also called the entry slip road and exit slip road, respectively. For ease of expression, the commonly used terminologies, i.e. “mainline”, “on-ramp” and “off-ramp” in the literature will be adopted in this study to describe the motorway.

rates, i.e. the number of vehicles entering the motorway mainline during each signal cycle (Arnold Jr 1998). For ease of expression, the example below only shows a typical scenario of ramp metering with only one controlled on-ramp. In some cases (such as the coordinated strategies), a ramp metering strategy may control more than one on-ramp.

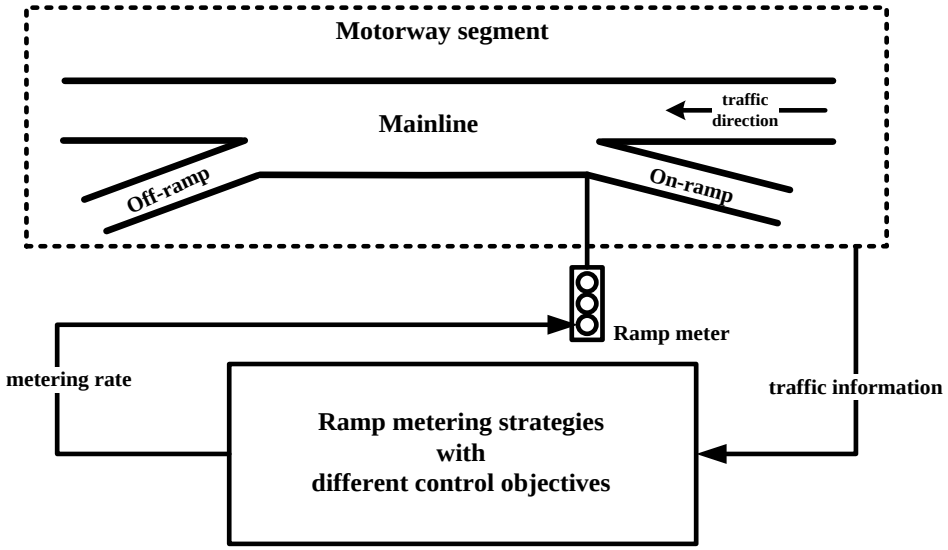


Figure 2.1: An example of ramp metering

Based on the traffic information (such as traffic flow, density and speed) collected from the motorway, a ramp metering strategy aims to generate suitable metering rates to alleviate traffic congestion and achieve some predefined control objectives such as reducing the total time spent on motorways, balancing the waiting time at on-ramps and decreasing vehicle emissions. In field applications, metering rates generated by the ramp metering strategy can be further converted to signal timings (i.e. the time duration of green, red and amber phases in each signal cycle) for ramp meters. According to different control strategies, the traffic information may be either the real-time traffic data collected from loop detectors (for traffic-responsive strategies) or the historical data recorded in the database (for fixed-time strategies). Different control objectives considered in the existing studies will be introduced in Section 2.2, while the control strategies using different traffic information will be reviewed in Section 2.3.

2.1.2 Traffic flow description

As mentioned above, traffic information regarding traffic flow, density and speed is essential for ramp metering strategies to generate metering rates. Hence, to better understand ramp metering problems, the background and some related terminologies of traffic flow should be known in advance. Generally, traffic flow has three fundamental characteristics namely flow, density and speed, which can be observed and described macroscopically or microscopically according to different levels of detail required (May 1990). The microscopic characteristics of road traffic are related to individual vehicles, such as the time and distance headway between two adjacent vehicles, and individual vehicle speeds. On the other hand, macroscopic information concentrates on traffic characteristics aggregated from a group of vehicles including traffic flow rates, density rates and average speeds. In the ramp metering area, macroscopic characteristics of traffic flow are usually used to describe road traffic and develop control strategies, which will be the focus of this study. The terminologies “flow”, “density” and “speed” mentioned in the following parts of this thesis all refer to their macroscopic descriptions, i.e. flow rate, density rate and average speed. These three terminologies are defined as follows (May 1990):

- Flow (expressed by q) is the number of vehicles passing a fixed point on the road during a period of time (usually one hour).
- Speed (denoted by v) is the average rate of motion of vehicles, which is expressed by distance per unit time (such as km/h). According to different observation methods, speed can be further divided into two classes: time-mean-speed and space-mean speed. Time-mean-speed is the measured average speed of vehicle groups that pass a fixed point on the road. Space-mean-speed is the given distance travelled by vehicles divided by the average time these vehicles spend on travelling.

- Density (represented by ρ) is the number of vehicles occupying a length of the roadway (usually one mile or one kilometre).

The basic relationship among these three characteristics is given by:

$$\rho = \frac{q}{v} \tag{2.1}$$

Through statistical studies of traffic data, more static relationships of flow, density and speed can be obtained through mathematical descriptions. As shown in Figure 2.2, one of the early works conducted by Greenshields (Greenshields et al. 1935) presented relationships between each two parameters. For a theoretical description, space-mean-speed instead of time-mean-speed is used here. Among these three diagrams, (a) presents the relationship between density and flow, (b) is the relationship between speed and density, and (c) shows the relationship between speed and flow. These three figures are usually named fundamental diagrams.

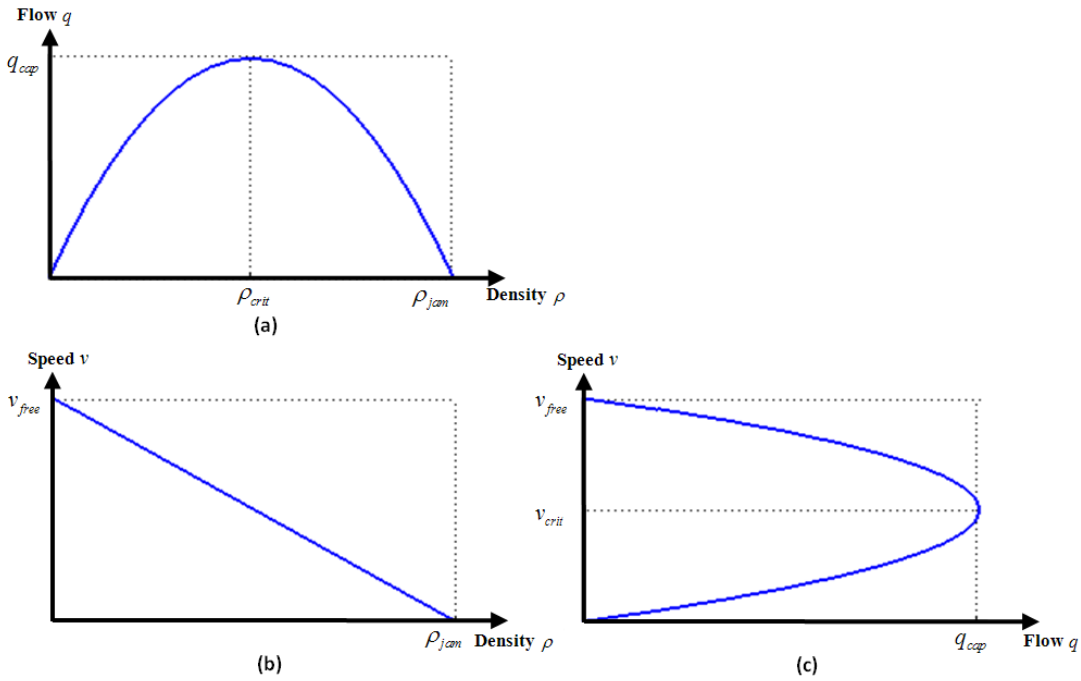


Figure 2.2: Fundamental diagrams

From fundamental diagrams, some important parameters can be used to describe the state of traffic under different conditions. These parameters and related traffic states are summarised as follows.

- q_{cap} denotes the road capacity that is the maximum number of vehicles passing a fixed point on the road during a period of time (usually one hour).
- ρ_{crit} is the critical density which corresponds to the road capacity q_{cap}. From Figure 2.2 (b) it can be seen that the critical density can divide traffic flow into two regimes. When traffic density is below the critical density, traffic flow is in the free-flow state. On the other hand, when traffic density exceeds the critical density, traffic flow is in the congestion state.
- ρ_{jam} is the jam density that dictates the maximum number of vehicles staying on a length of road. When jam density is measured, vehicle speed will drop to 0 and no flow can pass through.
- v_{free} is the free-flow speed. This speed is the average speed of vehicles that can move at their desired speed under the low traffic density situation.
- v_{crit} is the critical speed. Similar to critical density, this value corresponds to the maximum traffic flow, i.e. road capacity.

Fundamental diagrams are very important for analysing traffic flow models and traffic control strategies. After the contribution from Greenshields, some studies described fundamental diagrams as other shapes such as triangle and trapezoid. These diagrams will be explained in more detail when they are mentioned in other parts of this thesis.

2.2 Control Objectives

In the early stages, improving motorway traffic efficiency in terms of reducing total time spent by road users is the only concern of developing a ramp metering control strategy. However, some important social and environmental impacts such as user equity (i.e. equally allocating motorway resources to different users), vehicle emissions and user safety are neglected by efficiency-orientated strategies. In recent years, these impacts of ramp metering have attracted considerable attention and some of them have been introduced as additional control objectives to the ramp metering problem. In this section, different control objectives of ramp metering are briefly introduced.

2.2.1 Improving Efficiency

The traffic efficiency of a motorway system is usually regarded as the primary objective of a successful ramp metering strategy. High efficiency means that motorway users can spend as short a time as possible travelling on motorways (Kotsialos and Papageorgiou 2001). Generally, total time spent (TTS) by users on motorways is used as an indicator to measure efficiency. Improving efficiency is equivalent to reducing TTS. As suggested by (Papageorgiou and Kotsialos 2002), the TTS of a motorway segment can be expressed by:

$$TTS = T \cdot \sum_{k=0}^{N_k - 1} (n_{main}^k + n_{on}^k) \tag{2.2}$$

where, T is the time interval between two time steps, k is the index of time step, n_{main}^k and n_{on}^k denotes the number of vehicles on the mainline and on-ramp, respectively.

When improving traffic efficiency is the main concern, two main mechanisms may help ramp metering control strategies to achieve this goal. As summarised by (Gomes and Horowitz 2006, Papageorgiou and Kotsialos 2002), these two mechanisms are: (1) preventing mainline capacity drop, (2) increasing off-ramp outflows.

The capacity drop mentioned in (1) is a widely observed phenomenon in bottleneck locations on motorways, such as on-ramp merge areas with lane drops (Cassidy and Bertini 1999, Hall and Agyemang-Duah 1991). In the congested condition (the density exceeds the critical value), a queue forms on the mainline, and the maximum downstream flow (or queue discharge rate) of the bottleneck location will reduce suddenly, which will lead to a gap between the original road capacity and the queue discharge rate under the congested condition. When reducing the TTS by users is the main concern, this kind of capacity drop should be prevented by maintaining the traffic flow on the motorway mainline around the original road capacity.

The second mechanism is based on the fact that when the congestion propagates upstream and blocks the upstream off-ramps, the outflow of these off-ramps will be impeded. This will worsen congestion and increase delays on motorways. Therefore, easing the congestion in the mainline to avoid blocking off-ramps is necessary to improve traffic efficiency under the second mechanism. Two simple examples provided by (Papageorgiou and Kotsialos 2002) can be used to explain these two mechanisms clearly. In these two cases, all flows and demands are assumed to be constant during a period T.

(1) Preventing mainline capacity drop

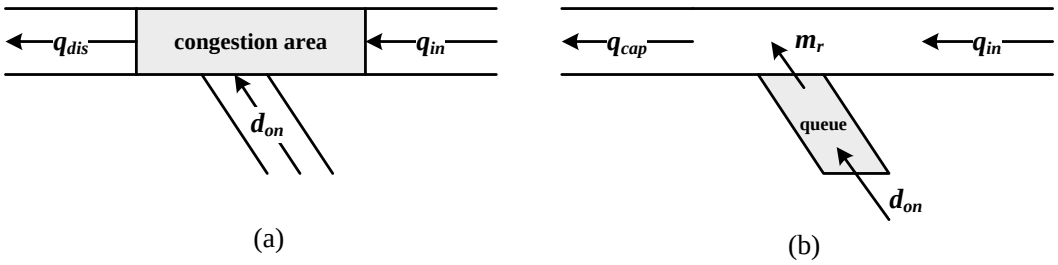


Figure 2.3: On-ramp only case: (a) no control, (b) with control (Source: Papageorgiou and Kotsialos, 2002)

The motorway segment considered in (1) only contains one on-ramp. As shown in Figure 2.3, q_{in} is the inflow entering the analysed motorway segment from its upstream segment, q_{dis} is the discharge flow under congestion conditions, d_{on} denotes the on-ramp demand, q_{cap} is the capacity, m_r is the on-ramp flow (or metering rate)[2] generated by control strategies. Assuming that $q_{in} + d_{on} > q_{cap}$, without control (Figure 2.3 (a)), congestion occurs on the motorway mainline and leads to capacity drop in the congestion area. In this condition, the outflow of motorway segment drops to the queue discharge rate q_{dis}. Thus, $n_{main} = q_{in} + d_{on} - q_{dis}$, $n_{on} = 0$ and the TTS can be calculated according to Equation (2.2), which is:

$$TTS^n = T(q_{in} + d_{on} - q_{dis}) \tag{2.3}$$

With control (Figure 2.3 (b)), the extra demand of on-ramp can be restricted from entering the motorway mainline. The metering rate calculated by the

2 The on-ramp flow and metering rate are not distinguished by the literature reviewed in this chapter (maybe for ease of expression) and are both indicated by m_r. However, the real on-ramp flow entering the mainline is not always the same as the generated metering rate (sometimes there may not be enough vehicles waiting at the on-ramp to reach the generated metering rate). In this study (Chapter 4), the on-ramp flow and metering rate will be indicated by two different variables m_r and c respectively.

control strategy should satisfy $m_r + q_{in} = q_{cap}$, which can maintain a high outflow of q_{cap} during the control period. Thus, $n_{main} = q_{in} + m_r - q_{cap}$, $n_{on} = d_{on} - m_r$ and the TTS in the controlled situation can be obtained from:

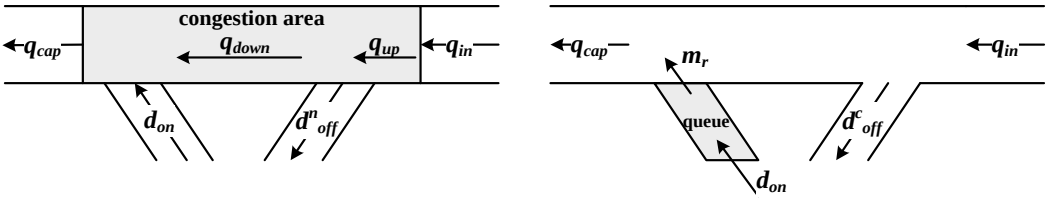
$$TTS^c = T(q_{in} + d_{on} - q_{cap}) \tag{2.4}$$

Therefore, compared with the non-controlled situation, the reduced TTS (in percentage) under control should be:

$$\Delta TTS = \frac{TTS^n - TTS^c}{TTS^{no}} \cdot 100 = \frac{q_{cap} - q_{dis}}{q_{in} + d_{on} - q_{dis}} \cdot 100 \tag{2.5}$$

For instance, if the mainline capacity drops by 5% ($q_{dis} = 0.95 \cdot q_{cap}$) and the demand is 20% more than capacity ($q_{in} + d_{on} = 1.2 \cdot q_{cap}$), then the TTS can be reduced by 20% with suitable control.

(2) Increasing off-ramp outflow



(a) (b)

Figure 2.4: Off-ramp including case: (a) no control, (b) with control (Source: Papageorgiou and Kotsialos, 2002)

The motorway segment with one on-ramp and one off-ramp is used for analysis in mechanism (2). d^n_{off} and d^c_{off} are outflows of off-ramps for non-controlled and controlled situations. $\beta \in [0,1]$ is used to denote the proportion of mainline flow that exits the motorway from its linked off-ramp. Thus, it can be obtained that: $d^n_{off} = \beta \cdot q_{up}$ and $d^c_{off} = \beta \cdot q_{in}$.

To distinguish the effects of two different mechanisms, mechanism (2) does not consider the capacity drop phenomenon, which means the outflow of the motorway mainline does not reduce and keeps at q_{cap} under the congestion conditions. Therefore, in the non-controlled situation, $q_{down} = q_{cap} - d_{on}$ and $q_{down} = q_{up} - d_{off}^{n}$ can be easily obtained from Figure 2.4 (a). Recall that $d_{off}^{n} = \beta \cdot q_{up}$. Thus, $d_{off}^{n} = [\beta / (1-\beta)] \cdot (q_{cap} - d_{on})$.

Similar to mechanism (1), the TTS for non-controlled and controlled conditions can be calculated by:

$$TTS^{n} = T(q_{in} + d_{on} - q_{cap} - d_{off}^{n}) \tag{2.6}$$

$$TTS^{c} = T(q_{in} + d_{on} - q_{cap} - d_{off}^{c}) \tag{2.7}$$

Considering $d_{off}^{c} = \beta \cdot q_{in}$ and $d_{off}^{n} = [\beta / (1-\beta)] \cdot (q_{cap} - d_{on})$, the reduced TTS by control strategies can be computed with:

$$\Delta TTS = \frac{TTS^{n} - TTS^{c}}{TTS^{n}} \cdot 100 = \beta \cdot 100 \tag{2.8}$$

If β is 0.05, the TTS will fall 5% compared with the non-controlled condition. Indeed, the larger β is, the more TTS can be reduced.

In some special cases that do not have off-ramps, preventing capacity drop is the only way to reduce the TTS, while in more general cases with both on- and off-ramps two mechanisms (1) and (2) may work together to improve efficiency.

2.2.2 Maintaining Equity

Although efficiency-orientated strategies have brought great benefits to the motorway system, these benefits may not be fairly allocated to all users involved, which causes severe inequity in using the motorway system

(Yafeng et al. 2004). The equity issue has been proposed as a negative impact of ramp metering since the 1960's (Pinnell et al. 1967), but it was usually neglected by ramp metering strategies. It was not until the recent decade that this issue began to attract enough attention. Highly inequitable strategies may lead to restricting access for some users to the motorway, while allowing others to enter the motorway freely. This problem has affected the public acceptance of using ramp metering and restricted the effects of ramp metering strategies (Yafeng et al. 2004). Under such circumstances, equity has been regarded as one of the main purposes for developing coordinated ramp metering strategies with network-wide applications (Papamichail et al. 2010).

User equity in a motorway system is defined as equally allocating motorway resources to different users from both a spatial and temporal point of view (Levinson and Zhang 2006, Levinson et al. 2002, Zhang and Levinson 2005). As summarized by Zhang and Levinson, user equity is classified into two categories, spatial equity and temporal equity. The former one measures the difference in time spent or travel speed among users from different on-ramps to the same motorway mainline at the same time. On the other hand, temporal equity means the equity among users who access the motorway mainline from the same on-ramp at different times. Compared with temporal equity, spatial equity is more meaningful in the network-wide scenario composed of multiple on-ramps. Thus, most of the existing control strategies are mainly focused on the spatial equity and its impacts on traffic efficiency (Kotsialos and Papageorgiou 2004a, Meng and Khoo 2010, Yafeng et al. 2004, Zhang and Levinson 2005). This study also concentrates on the spatial equity issue, and the term "equity" in the following sections only refers to the spatial equity.

Although some work has been done to develop advanced algorithms to deal with equity problems, the measurement for capturing the equity is not well studied. Each study has its own indicator to measure user equity, and explains the equity in different ways. Some measurements used to capture the equity of a ramp metering system are summarised here.

Gini coefficient

$$\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\left|d_i - d_j\right|}{2N\sum_{i=1}^{N} d_i} \tag{2.9}$$

where d_i and d_j are delays of users i and j ($i \neq j$), N is the number of users. The Gini coefficient is a commonly used index in economics to measure the inequity of incomes, which is derived from the so-called Lorenz curves. This index was first used by Levinson et al. (2002) to measure the equity of a ramp metering system. However, this index needs the information of each individual vehicle on motorways which is difficult to obtain in real time and not suitable for developing real-time traffic control strategies. Another similar index using the Gini coefficient is proposed in (Yafeng et al. 2004), where the time saving ratios instead of user delays were used to form the index. For the same reason as for measurement (2.9), this index cannot be used for algorithm development.

Spatial variance of travel times

$$\frac{\sum_{i=1}^{N}(\bar{T}^k - T_i^k)^2}{N} \tag{2.10}$$

where T_i^k is the average travel time of users for queuing at on-ramp i and travelling a fixed length (6.5 km in their work) of the motorway mainline, $\bar{T}^k = \sum_{i=1}^{N} T_i^k / N$ and N is the number of on-ramps, k is the time step. This

measurement was used by Kotsialos and Papageorgiou (2004a) to measure the impact of their strategy on equity, but the measurement itself was not a part of the proposed control algorithm. In their algorithm, equity was considered by setting constraints for on-ramp queues.

Total weighted travel time

$$TWTT = WFTT + WRD \tag{2.11}$$

where, $TWTT$ is the total weighted travel time of the motorway network, $WFTT$ is the weighted mainline travel time, WRD is the weighted on-ramp delays. Although Levinson et al. (2002) have used the Gini coefficient to measure the equity of a ramp metering system, it cannot be directly used to develop a control algorithm. In their following study that focused on developing a control algorithm with consideration of equity, another measurement, the weighted total travel time was used (Zhang and Levinson 2005). The aim of this objective function is to balance the efficiency and equity through minimising the total weighted travel time.

Ratio of minimum and maximum delays

$$\frac{\min\left\{\sum_k D_i^k\right\}}{\max\left\{\sum_k D_i^k\right\}}, \quad i = 1, 2, ..., N \tag{2.12}$$

where, D_i^k is the average delay of users from on-ramp i at step k. The measurement (2.12) uses the ratio of the minimum and maximum average delays of N different on-ramps to capture the equity (Meng and Khoo 2010). This ratio should be between 0 and 1. The user equity will increase when this ratio approaches to 1. This equity index was incorporated as a part of the objective function of their control strategy.

Equity constraint

$$\frac{m_{r,1}^k}{d_{on,1}^k} = \frac{m_{r,2}^k}{d_{on,2}^k} = \cdots = \frac{m_{r,N}^k}{d_{on,N}^k} \tag{2.13}$$

In the work presented in (Zhang and Wang 2013), the ratio of metering rate m_r^k and demand flow d_{on}^k at each time step k was used to measure the equity. In their study, a good equity was obtained by making all the ratios of different on-ramps equal to each other.

Some other work such as (Zhang and Shen 2010) and (Tian et al. 2012) formulated the equity problem of ramp metering from a more theoretical point of view that was based on a so-called monocentric network (without off-ramps) and did not consider the inside queues. Although some mathematical proofs and analysis work can be conducted in the simplified case, many realistic networks (with off-ramps) do not fit these assumptions. This kind of theoretical work is not the scope of this thesis.

Of the equity measurements mentioned above, only the last three measurements, i.e. (2.11), (2.12) and (2.13) were used to develop equity-related control algorithms. The main problem of measurement (2.12) is that it only captures the minimum and maximum on-ramp delays, and it cannot measure any in-between values when more than two on-ramps are included. Thus, as long as two cases have the same minimum and maximum on-ramp delays, they will have the same equity. There is a lack of clear definition of equity in (Zhang and Wang 2013), and why the ratio of metering rate and on-ramp demand flow in measurement (2.13) can express the equity is not explained. Compared with (2.12) and (2.13), measurement (2.11) is more reasonable, which aims to balance the efficiency and equity by assigning weight values to the mainline travel time and on-ramp delays. In the study presented in this thesis, a similar concept of measurement (2.11) has been

used to derive the rewards (related to control objectives) of a self-learning system. The detailed definition of these rewards will be introduced in Chapter 4.

2.2.3 Other objectives

Except for efficiency and equity, some other control objectives such as reducing vehicle emissions and increasing user safety were also considered for ramp metering recently.

Compared with efficiency and equity, how to reduce vehicle emissions by ramp metering has not been well studied. It is only until recent years this issue has been considered as one additional control objective for ramp metering (Csikós and Varga 2012, Zegeye et al. 2012). By combining vehicle emission models with a dynamic traffic flow model, these strategies aim not only to decrease the total time spent by users on motorways, but also to reduce vehicle emissions in the meanwhile. Another effect of ramp metering is to improve user safety through reducing the number of accidents on motorways (Abdel-Aty et al. 2007, Bhouri et al. 2013, Lee et al. 2006). These studies have shown that user safety is related to ramp metering. However, very few of them focused on developing control strategies specific to solving safety problems. Most work related to this issue was to evaluate the effect of existing ramp metering strategies on improving user safety on motorways (Abdel-Aty et al. 2007, Bhouri et al. 2013, Lee et al. 2006).

Although vehicle emissions and user safety have been proposed as additional concerns of ramp metering, how to incorporate them into a control strategy has still not been well studied. One important reason for that is, except for the basic traffic flow operation, additional models are required to estimate vehicle emissions and accident potential when making the ramp metering strategies. Developing these models that can explicitly measure

emissions and safety itself is already a challenge in real-time situations (Banks 2000, Lee et al. 2006). In this study, the main focus is on the basic traffic flow operation and related objectives, namely traffic efficiency and user equity.

2.3 Control Strategies

The last section gave a brief introduction to control objectives, which has shown what can be achieved by ramp metering. This section focuses on how to achieve these objectives through specific control strategies. Traffic control strategies can be roughly classified into two categories: fixed-time strategies and traffic-responsive strategies (Papamichail et al. 2010, Zhang and Wang 2013). Fixed-time strategies are also known as pre-timed strategies, which are based on the constant historical data and adopt an off-line method. Section 2.3.1 will give a brief introduction to this approach. Traffic-responsive strategies provide control solutions with consideration of dynamic traffic conditions and real-time measurements from the road network (Papamichail et al. 2010). These strategies can be further classified into two categories, namely local strategy and coordinated strategy according to their working scope (Papageorgiou et al. 2003, Papamichail et al. 2010), which will be introduced in Sections 2.3.2 and 2.3.3 respectively. The RL-based method can be regarded as one traffic-responsive strategy according to its capability of responding to traffic dynamics in real time. As the RL-based method is the main focus of this thesis, this method will be introduced in a separate section (Section 2.3.4). For each kind of strategy mentioned above, two control objectives, traffic efficiency and user equity will be highlighted.

2.3.1 Fixed-time strategies

As one early attempt tackling ramp metering problems, Wattleworth (1965) modelled ramp metering for a motorway system as an optimisation problem according to historical demands for different times of day. This model is shown in (2.14), where the motorway studied was divided into several segments, and each segment contained one on-ramp.

$$\begin{aligned} &\max \sum_i m_{r,i} \\ &\text{s.t.} \quad \sum_i x_{ij} m_{r,i} \le q_{cap,j}, \forall j \\ &\qquad 0 \le m_{r,i} \le d_{on,i}, \forall i \end{aligned} \tag{2.14}$$

where, $m_{r,i}$ is the on-ramp flow of the i th motorway segment, x_{ij} is the proportion of vehicles that came from the on-ramp of segment i and passed through segment j, $q_{cap,j}$ is the mainline capacity of segment j, $d_{on,i}$ is the demand flow rate for the on-ramp of segment i. By setting suitable constraints for motorway mainline traffic flows ($\sum_i x_{ij} m_{r,i} \le q_{cap,j}$) and on-ramp traffic flows ($0 \le m_{r,i} \le d_{on,i}$), they aimed to maximise total on-ramp flows entering the mainline. It was equivalent to improving traffic efficiency.

Besides the basic constraints shown in Equation (2.14), they suggested two more constraints to restrict the number of vehicles waiting at on-ramps. The first constraint is $n_{on,i} \le n_{on}^{\max}$, which means the on-ramp queue $n_{on,i}$ of on-ramp i should be less than a predefined boundary $n_{on}^{\max}$. This boundary can be set according to the maximum on-ramp storage space. The second restriction is expressed by $n_{on,i} = n_{on,i+1}$ that guarantees the uniform distribution of queues at different on-ramps. This constraint is set for the requirement of user equity on motorways.

After the work carried out by Wattleworth, some similar objective functions and constraints were proposed for different control aims, such as balancing

on-ramp queues, maximising total travel distance and total vehicular input (Yuan and Kreer 1971, Wang and May 1973, Chen et al. 1974, Schwartz and Tan 1977). Techniques such as linear programming were used to solve these optimisation problems in an off-line situation.

Considering the evolution of traffic flow and variable demands, Papageorgiou (Papageorgiou 1980) proposed a more accurate model as an extension of static models mentioned above. In this work, the control period was divided into several intervals with different on-ramp demands. By introducing constant travel time for each motorway segment, the new model reformulated the static model in terms of outflow calculations for each segment, objective function and related constraints. This model can also be solved by linear programming.

Fixed-time strategies can find the optimal metering rates according to static demands and fixed traffic conditions for different times of day. However, demands are not constant and may fluctuate within the control period. For the same time on different days, demands may also be different because of some special events such as incidents, road work and weather conditions (Papageorgiou and Kotsialos 2002). As fixed-time strategies do not take account of the real-time demands, they may make the motorway overloaded or underutilised.

2.3.2 Local traffic-responsive strategies

To overcome the limitations of fixed-time strategies, traffic-responsive strategies are developed to respond to the real-time traffic dynamics. Local traffic-responsive methods only measure the traffic information within the vicinity of one controlled motorway segment, and do not consider the performance of the whole motorway network. Thus, most local methods only focus on efficiency improvements within its controlled range. The equity

problem which needs the network-wide information from other on-ramps is neglected.

Feed-forward control

One of the most popular local strategies is called the demand-capacity strategy, which adopts an open-loop (feed-forward) control method (see Figure 2.5).

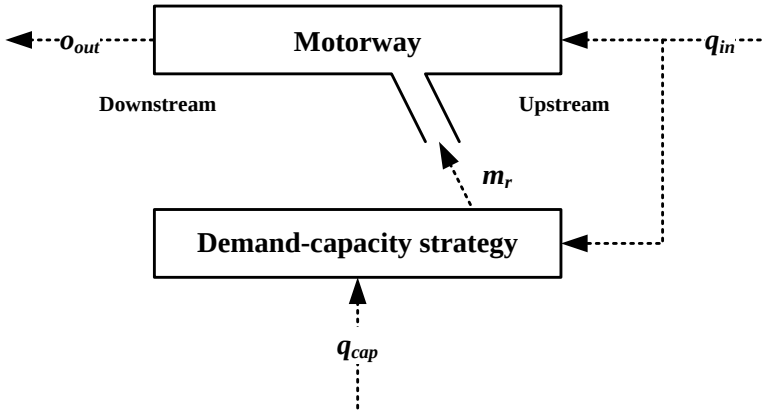


Figure 2.5: Demand-capacity Strategy
(Kotsialos and Papageorgiou 2004b)

At each time step, the metering rate can be calculated using the following equation (Masher et al. 1975) :

$$m_r^k = \begin{cases} q_{cap} - q_{in}^{k-1}, & \text{if } o_{out}^k \le o_{crit} \\ m_r^{\min}, & \text{otherwise} \end{cases} \tag{2.15}$$

where, m_r^k is the on-ramp flow (or metering rate) calculated at time step k, q_{cap} is the motorway capacity collected downstream of the on-ramp, q_{in}^{k-1} is the flow entering the controlled segment from the upstream motorway at time step $k-1$, o_{out}^k is the occupancy measured downstream of the on-ramp at time step k, o_{crit} is the critical occupancy of downstream motorway, and $m_r^{\min}$ denotes the minimum metering rate that is a predefined value. Usually, there exists a relationship between the occupancy and flow, and critical occupancy o_{crit} corresponds to the maximum flow (capacity) q_{cap}. The main

objective of the demand-capacity strategy is to keep the flow departing the controlled motorway segment (measured downstream) close to the predetermined capacity. Another similar method also following the feed-forward mechanism is the occupancy strategy (Masher et al. 1975) which adopts occupancy instead of flow q_{in} to calculate metering rates.

Feedback control

ALINEA (Asservissement Linéaire d'Entrée Autoroutière, i.e. Linear feedback control of a motorway on-ramp) is a widely used local ramp metering strategy, which is based on the feedback control mechanism as shown in Figure 2.6. Under the structure of ALINEA, the system output can be used to regulate the input.

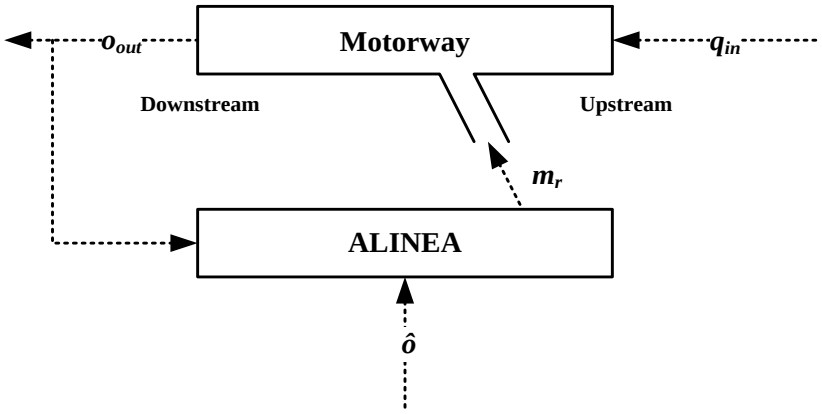


Figure 2.6: ALINEA Strategy
(Kotsialos and Papageorgiou 2004b)

The ramp metering rate at each time step is given by Equation (2.16) (Papageorgiou et al. 1991).

$$m_r^k = m_r^{k-1} + K_R(\hat{o} - o_{out}^k) \tag{2.16}$$

where, K_R is a regulatory parameter ($K_R > 0$), $\hat{o}$ is a predefined target value which is typically set as a value close to the critical occupancy o_{crit}. In this way, ALINEA has the same objective of demand-capacity strategy that

is to keep the motorway outflow close to the capacity. By using the demand-capacity strategy, the metering rate will be reduced to the minimum value after the outflow exceeds the road capacity ($o_{out}^{k} \le o_{crit}$), which is relatively rough and unstable. On the other hand, ALINEA regulates the metering rate more smoothly and can avoid congestion in a more stable way.

Besides the original ALINEA algorithm which uses downstream occupancy as the control variable, a number of variations, namely FL-ALINEA, UP-ALINEA and UF-ALINEA, using different control variables such as downstream outflow, upstream occupancy and upstream inflow under the same control logic were also developed by the same authors (Smaragdis and Papageorgiou 2003).

In practical applications, the on-ramp storage space is an important issue that should be considered by ramp metering strategies. When the on-ramp queue exceeds its storage space, it may spill back onto the adjacent local streets and cause severe congestion. Thus, a successful ramp metering strategy should be able to constrain the on-ramp queue under a maximum permitted value. By combining a queue management algorithm, ALINEA can be extended to ALINEA/Q which can take the queue constraints into account (Smaragdis and Papageorgiou 2003). This queue management algorithm is given by:

$$n_{on}^{k+1} = n_{on}^{k} + T(d_{on}^{k} - m_{r}'^{k}) \tag{2.17}$$

$$m_{r}'^{k} = -\frac{1}{T}(n_{on}' - n_{on}^{k}) + d_{on}^{k-1} \tag{2.18}$$

where, n_{on}^{k} is the on-ramp queue length (the number of vehicles) at step k and n_{on}' is the queue constraint, T is the time interval between two steps. Equation (2.17) is used to estimate queue length, and equation (2.18)

determines the maximum metering rate that can keep queue length under constraint.

By considering the metering rates generated by original ALINEA (m_r^k) and queue management algorithm ($m_r'^k$), the final metering rate m_{fr}^k is determined by:

$$m_{fr}^k = \max\{m_r^k, m_r'^k\} \tag{2.19}$$

Through Equation (2.19), ALINEA/Q can make the ramp queue avoid exceeding queue constraints when it tries to achieve its target, such as keeping mainline density around the critical value.

Other approaches

Except for algorithms using the classic control theory, such as demand-capacity (feed-forward control) and ALINEA (feedback control), some other approaches from artificial intelligence such as neural networks and iterative learning were also used to develop local ramp metering strategies.

For instance, a neural network was combined with a feedback controller in (Zhang and Ritchie 1997) to maintain road density around the critical density. For the same purpose, an iterative learning algorithm was developed by (Hou et al. 2008). The main drawback of these methods is similar to algorithms based on the classic control theory, i.e. a target value such as critical density (occupancy) or capacity flow should be defined in advance. As mentioned in (Papamichail et al. 2010) this target value (capacity flow) is not stable in some realistic situations, and any algorithms trying to obtain a predefined value may worsen the traffic operations. Moreover, without suitable optimisation mechanism, these methods cannot be easily extended to solve multi-objective problems. Therefore, these methods are usually

limited to local problems with one control objective that is to improve traffic efficiency.

2.3.3 Coordinated traffic-responsive strategies

Unlike the local methods introduced above, coordinated strategies take the whole network into account and all involved on-ramps can be uniformly controlled by the coordinated ramp metering strategy. With network-wide information, equity issues can be considered by the coordinated strategies. The remainder of this section will review some important algorithms and strategies in this area.

Efficiency-orientated strategies

Although equity has been proposed as one important impact of coordinated ramp metering strategies (Papamichail et al. 2010), many of them are still only focused on efficiency improvement. One example of these efficiency-oriented strategies is known as METALINEA (Papageorgiou et al. 1990), which is an extension of the local strategy ALINEA. METALINEA followed the feedback control logic, and tried to generate metering rates for all controlled on-ramps simultaneously. To calculate different metering rates for different on-ramps, METALINEA used a number of vectors and matrices as shown in the equation below.

$$\mathbf{m}_{\mathbf{r}}^{k} = \mathbf{m}_{\mathbf{r}}^{k-1} - \mathbf{K}_{\mathbf{1}}(\mathbf{o}^{k} - \mathbf{o}^{k-1}) + \mathbf{K}_{\mathbf{2}}(\hat{\mathbf{O}} - \mathbf{O}^{k}) \tag{2.20}$$

where, $\mathbf{m}_{\mathbf{r}}$, $\mathbf{o}$, $\hat{\mathbf{O}}$ and $\mathbf{O}$ are all vectors and each vector contains a group of elements related to different on-ramps, $\mathbf{K}_{\mathbf{1}}$ and $\mathbf{K}_{\mathbf{2}}$ are two gain matrices (containing a group of regulatory parameters) that should be calibrated for specific problems. The objective of METALINEA is the same as ALINEA that is to keep the road density or occupancy at a predefined level. Besides the feedback control method, some other operational algorithms such as FLOW

(Jacobson et al. 1989), SWARM (Paesani et al. 1997) and ZONE (Lau 1997) also attempted to improve traffic efficiency by maintaining the outflow around the road capacity.

Another group of studies focused on formulating different ramp metering scenarios as optimisation problems and using optimal control strategies to solve them. The purpose of these strategies was to optimally solve an efficiency-related objective function, not achieve some predefined target value. Examples of optimisation-based methods can be found in (Zhang et al. 1996, Zhang and Recker 1999, Gomes and Horowitz 2006, Chow and Li 2014), where macroscopic traffic flow models were combined with control strategies to formulate optimisation problems. Traffic dynamics generated by these traffic flow models at each time step can be involved in the optimisation process, which is different from the fixed-time strategies introduced in Section 2.3.1.

Equity-involved strategies

In recent two decades (especially the recent ten years), the equity issue has attracted increasing attention which has been considered in some coordinated ramp metering strategies.

One of the early attempts was made by (Benmohamed and Meerkov 1994), where a feedback control framework for equity consideration was developed. This control method adopted a decentralised architecture that balanced the local efficiency (benefits for each section) and global equity (benefits for all sections involved). In their work, the so-called max-min fairness[3] was

---

[3] The max-min fairness is an objective about the resource allocation in a communication network, which can be achieved when improving the resource (such as bandwidth) obtained by any one user of a specific user group will lead to the decrease of resources allocated to some other users in the same group.

guaranteed in a hypothetical freeway network. For motorway traffic control, the recourse allocation problem was transformed to the road capacity allocation problem. Thus, the fairness in a freeway system was obtained by equally distributing the available road capacity to users on different routes between different OD pairs.

Kotsialos et al. (2001) developed an optimal coordinated control method named advanced motorway optimal control (AMOC) for large-scale situations. By adding some constraints to the objective function to restrict the queue length on each on-ramp involved, they showed in their work (Kotsialos and Papageorgiou 2001, Kotsialos and Papageorgiou 2004a) that the equity problem can be partially tackled, if the equity was measured by Equation (2.10). By considering the uncertainty of the motorway traffic system, a hierarchical control approach composed of AMOC and ALINEA were developed in (Papamichail et al. 2010), which provided a similar performance of AMOC on maintaining equity using the same measurement.

Without an explicit equity measurement, the infrastructure limitations (maximum queue length on on-ramps) are not enough for maintaining equity of the whole system. Zhang and Levinson (2005) proposed a more explicit way using an equity-related control objective (the measurement (2.11)) to balance efficiency and equity of motorway systems. The main contribution of their work is the weighted total travel time that combines the efficiency and equity problem into one objective function. Different weight values need to be defined according to different on-ramp delays. For the equity consideration, larger weight values should be assigned to longer on-ramp delays to encourage more vehicles to enter the mainline. The goal of their control algorithm is to minimise this weighted total travel time by coordinating a number of on-ramps in the upstream of the critical section (the congested motorway segment).

However, prior knowledge is required in the above strategy to predefine different weight values, which is usually very difficult to obtain by system designers. Thus, Meng and Khoo (2010) formulated a multi-objective optimisation problem for balancing efficiency and equity. Without defining any explicit weights for any objective involved, a set of non-dominated solutions forming a so-called pareto front [4]can be obtained by solving a multi-objective optimisation problem. In this problem, the equity measurement (2.12) was used as one of the control objectives. For the real application, operators in the control room can choose their preferred solutions from the solution set.

Considering the computational complexity of mathematical models in the existing optimal control strategies, Zhang and Wang (2013) presented a hierarchical control method based on linear programming. Three objective functions aiming to minimise congestion, restrict the on-ramp queue length without spilling back, maximise throughput and balance the equity (the last two objectives are combined into one objective function with equity constraints shown in (2.13)) were formulated and assigned with different priorities. Through linear programming, these functions can be solved efficiently.

Summary

It can be seen from the review of recent studies that optimisation-based methods (such as optimal control) have attracted considerable attentions, because these strategies are theoretically sound and can be easily extended to deal with different control objectives. Many of existing equity-involved

---

[4] Pareto front is composed of non-dominated solutions. These solutions cannot be updated by improving any objectives considered without degrading at least one of the other objectives.

strategies fall into this category. However, most of them are model-based methods that cannot adapt to different simulation models without modifications. For those strategies based on complicated mathematical models, high computational demand is required to solve the optimisation problem, which increases the difficulty of field application (Jacob and Abdulhai 2010). To overcome these limitations, another optimisation-based method, reinforcement learning (RL), was introduced to the ramp metering area. This method is based on the Markov decision process and dynamic programming, which can approximately solve the optimisation problem through continuous learning without any models. Some recent applications of RL are investigated in the following section.

2.3.4 Reinforcement learning based strategies

RL-based control strategies for ramp metering are still in the early stages and most of them only have a local view with efficiency improvement as the main control objective. Examples of RL strategies from both local and coordinated perspectives are shown below.

Local RL-based strategies

The first work on the use of RL to solve ramp metering problems was conducted by Jacob and Abdulhai (Jacob and Abdulhai 2006, Jacob and Abdulhai 2010). In their system, ramp metering was combined with VMS to deal with incident-induced congestion and the coordination of multiple ramp meters was not considered. This approach was based on RL or specifically Q-learning which adopted a “trial-and-error” method to improve the strategy for selecting control actions (metering rates and VMS settings) from a predefined action set. After a number of trials, the best control actions under different traffic situations can be obtained to reduce the total time spent by users. In this system, the reward at each time step was derived from the

measurement of TTS shown in Equation (2.2). A six-dimensional state space was used by the control system, which contained two sub-sets for speed (for two roads), one sub-set for ramp meter (metering rates), one sub-set for VMS (VMS message settings) and two sub-sets for incidents (for two roads).

Except for the main concern of maximising motorway throughput (or mainline outflow which is equivalent to minimising TTS), on-ramp queue constraints were considered in (Davarynejad et al. 2011). This system also adopted Q-learning to solve local ramp metering problems. To reduce the computational complexity, an additional agent (or controller) was developed specifically for managing on-ramp queues in this system. The final metering rate was determined by the maximum value generated by two controllers responsible for maximising throughput and regulating on-ramp queues respectively. In this work, the state space was composed of five sub-sets regarding the downstream density, on-ramp queue length, metering rates, current and one-step predicted on-ramp demand. Another similar Q-learning based algorithm was presented in (Wang et al. 2012) which also tried to maximise the mainline outflow. The reward of their system was directly related to the mainline outflow, while the mainline inflow and ramp metering rates were used to form the state space.

Rezaee et al. (2012) used the Q-learning algorithm with function approximation to deal with the ramp metering control problems under the continuous state space. In this work, the state space was not composed of a number of discrete states as in the traditional Q-learning problems, but all possible states observed from the external environment. With the help of an algorithm based on k-nearest neighbours, the Q values (related to control objectives) for new observed states can be estimated. Similar to (Jacob and Abdulhai 2006, Jacob and Abdulhai 2010), the reward in this work was derived from the Equation (2.2) for TTS reduction. It was found in this work

that TTS was reduced by 44% compared with non-controlled situation. Two states regarding the mainline density and on-ramp flow were used to form the state space. In one of the later works shown in (Rezaee et al. 2013), some analysis and suggestions about parameter settings under continuous state conditions were provided. In another work presented in (Rezaee et al. 2014), the comparison of different function approximation methods were discussed.

Another local RL-based system was proposed in (Lu et al. 2013). This system extended the basic Q-learning to deal with incident-induced congestion. This method combined the direct reinforcement learning (DRL, i.e. the basic Q-learning) and the model-based planning together to obtain the benefits from both sides. The new method was compared with DRL and ALINEA. Experimental results obtained from simulation showed that, with suitable weight values, IRL can achieve a superior performance in many scenarios. Moreover, compared with DRL, IRL has a faster learning speed. The detailed description of this extended system will be given in Chapter 8.

Coordinated RL-based strategies

Besides the local applications, some recent studies have been done to explore the coordinated application of RL. In the work presented in (Bai et al. 2009, Zhao et al. 2011), a new coordinated method based on adaptive dynamic programming (ADP) was used to control a hypothetical motorway with four pairs of on- and off-ramps where ADP is a practical implementation and extension of RL (Lewis and Vrabie 2009). Compared with basic RL, two extra networks namely critic network and action network are maintained in the ADP structure. The critic network is used to generalise the reward function with respect to states, while the action network can correlate actions and rewards. In this way, continuous states and actions can be considered in the ADP-based method, which makes the control system more accurate

than the discrete RL. However, both the critic and action networks should be trained before or during the basic learning process, which requires extra computation and complicates the whole system. Similar to AMOC analysed in (Kotsialos and Papageorgiou 2004a), although queue constraints were considered in the ADP system to prevent on-ramp queues from exceeding their maximum permitted values, there was no equity measurement to capture equity conditions. Thus, the equity issue cannot be explicitly solved in this work.

Another RL-based coordinated ramp metering system was proposed by (Veljanovska et al. 2010, Veljanovska et al. 2012). Although some positive simulation results for improving traffic efficiency were shown in their work, the whole system in terms of three critical elements (i.e. state, action and reward) and how multiple agents (responsible for controlling multiple on-ramps) worked with each other was not clearly defined. It was mentioned in this work that the objective was to maximise the exit flows (including the outflows of both the mainline and off-ramps), but how this objective was converted to the reward at each time step was not presented. The same problem occurred with the action definition, and it was not clear what kinds of actions (such as how many vehicles are allowed to enter the mainline at each control step) were adopted.

One of the most recent applications of coordinated RL in the ramp metering area is (Fares and Gomaa 2015). In their work, the multi-agent concept based on the coordination graph was used to build a learning system. Similar to the feedback control algorithm ALINEA, the goal of this system was to keep the mainline density close to the critical value. Through simulation experiments, they showed that the new system can significantly reduce the travel time of road users. However, the equity issue was not considered by this system.

In the above three coordinated RL systems, the equity issue was not solved, and improving traffic efficiency was the only objective considered. Zhaohui and Kaige (2010) developed a new system based on RL that can take the equity problem into account. In their work, equity was measured by Gini coefficient and calculated at each time step. As mentioned in Section 2.2.2, to obtain the Gini coefficient, the relevant information (such as speed and travel time) of each user at each on-ramp is required. This information is very difficult to obtain in a real-time situation. It was not clear in this work how this information can be captured by the control system and converted to rewards at each time step.

2.4 Summary and Discussion

Through a brief investigation of control objectives, it can be seen that traffic efficiency is not the only concern of ramp metering in many recent studies. Increasing impacts from social and environmental sides, especially the equity issue, have been considered by the ramp metering community. To maintain user equity in a motorway system, some ramp metering strategies that can balance waiting times (or delays) for users from different on-ramps have been developed.

The review of ramp metering strategies showed that, compared with fixed-time methods, traffic-responsive approaches are more effective under dynamic conditions and have become the major strategies for ramp metering. Among traffic-responsive strategies, an increasing trend of using optimisation-based methods such as optimal control was shown in the literature. This is mainly because these strategies can solve the ramp metering problem based on optimisation theory that can provide sound solutions.

As an approximated optimisation method, RL was recently proposed to overcome some drawbacks of traditional optimal control strategies, such as poor adaptability and high computational demand. However, existing studies using RL are still in their early stages and have shown some limitations which are summarised as follows.

(1) There is a lack of a general framework for designing RL-based system for a ramp metering application, and each study has its own way to define RL elements, especially the state and reward. For example, the rewards in (Jacob and Abdulhai 2010, Rezaee et al. 2012, Rezaee et al. 2013) were derived from a commonly used TTS measurement, while in (Davarynejad et al. 2011, Wang et al. 2012) the rewards were related to outflows of the controlled motorway. Besides the reward, variant definitions of state space can be found in existing studies. Different states were used to form different state spaces from the simple 2-dimensional state space such as (Rezaee et al. 2012, Wang et al. 2012) to the more complicated 6-dimensional state space (Jacob and Abdulhai 2010). There was not enough explanation in these studies as to why these states were chosen. Some of the existing studies, such as (Veljanovska et al. 2012, Veljanovska et al. 2010, Zhaohui and Kaige 2010) even missed a clear definition of these elements, which makes their strategies very difficult to understand and be generalised to different scenarios.

(2) Although a few studies have considered the coordination problems in a RL-based strategy, improving traffic efficiency is still the main concern. How to add new objectives such as user equity and balance different control objectives have not been well studied. According to the review presented in Section 2.3, only one work has been done to incorporate the equity issue into the RL-based system. However, this work used the

Gini coefficient as the equity measurement, which is difficult to obtain in real time (as discussed in Section 2.2.2). How this coefficient was captured by their system and how to convert it to the equity-related rewards was not mentioned.

(3) There is no systematic evaluation for the RL-based system regarding the influence of learning parameters and the effectiveness of algorithms on different traffic networks (hypothetical and real networks). Most of existing studies selected learning parameters on an ad hoc basis without analysing these parameters in advance. To the best knowledge of the author, the only published work related to the analysis of learning parameters for ramp metering is shown in (Rezaee et al. 2013). This work provides some useful suggestions about how to select suitable parameters in a continuous state case with some adaptive settings. However, the behaviour of different parameter values and their impacts on the algorithm performance are not analysed, especially in a more common case with discrete states. Moreover, only a few studies such as (Jacob and Abdulhai 2010, Rezaee et al. 2012) have evaluated the RL-based system using real traffic data collected from a real motorway network, and most of the existing studies were based on a hypothetical network with assumed traffic demands.

To overcome these limitations, the main objectives introduced in Chapter 1 can be explained as follows:

(1) To investigate the state of the art of RL technology and its applications in the ramp metering domain. This objective forms the basis of all the other four objectives. By achieving this objective, the basic mechanism of RL and ramp metering can be obtained to develop the self-learning control system. The review of RL-based ramp metering systems has been

accomplished in this chapter, and the investigation of RL mechanism will be presented in Chapter 3.

(2) To provide a general framework for designing a RL-based ramp metering system. This framework contains a definition of three RL elements, namely reward, state and action, in a general ramp metering scenario and a structure with related modules that can bring together three elements and accomplish the learning process. The details of these elements and modules will be presented in Chapter 4.

(3) To explore the application of RL to ramp metering for both single- and multi-objective problems under the framework proposed by (1). Two control objectives relating to the traffic efficiency and user equity are considered in this study. A specific reward will be defined for each control objective. After that, two control algorithms will be developed to deal with single- (only efficiency) and multi-objective (both efficiency and equity) problems respectively. This part of the system design will also be introduced in Chapter 4.

(4) To provide a platform with initial software implementations based on objectives (1) and (2), which can be used to evaluate the RL-based system. The proposed system and a macroscopic traffic flow model will be programmed as two reusable classes by C++. This provides a flexible way to evaluate the RL-based system under different traffic conditions simulated by the traffic flow model. The detailed implementation issue will be tackled in Chapter 5.

(5) To evaluate the proposed system based on (3) by conducting simulation-based experiments considering both hypothetical and real traffic networks. Three cases including two hypothetical cases and one real case (with the real traffic data collected from a real motorway network)

will be designed to evaluate the proposed system on different tasks, such as improving traffic efficiency, managing on-ramp queue length and maintaining user equity at different on-ramps. This part of evaluation will be presented in Chapters 6 and 7. Moreover, an extension of the proposed system will also be tested in a simulation environment where the ability of the new system to deal with incident-induced congestion will be analysed. This test will be shown in Chapter 8.

The background and basic mechanisms of RL will be introduced in the next chapter.

CHAPTER 3 REINFORCEMENT LEARNING

This chapter introduces the basic idea and mechanism of reinforcement learning (RL). Section 3.1 begins with a brief introduction to how the agent interacts with its external environment by using RL. Then, the theoretical basis of RL, in terms of the Markov decision process (MDP) and dynamic programming (DP) are introduced in Sections 3.2 and 3.3. Section 3.4 gives the core algorithms, i.e. temporal difference (TD) algorithms of RL. The extension of basic RL algorithms to multi-objective problems is presented in section 3.5. The summary of this chapter is given in Section 3.6.

3.1 Agent and Environment Interaction

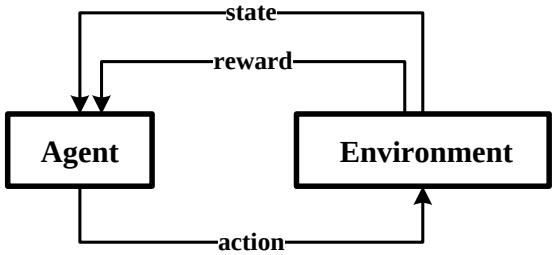


Figure 3.1: Agent and environment interaction

In a RL problem, the learning process is conducted through the interaction between an agent and its external environment as shown in Figure 3.1. Here, the agent is defined as an autonomous entity that can observe the environmental changes and take actions in response (e.g. the controller that can generate suitable metering rates in a ramp metering problem). The environment can be anything that should be controlled by the agent and is usually represented by a group of states (e.g. the state of traffic flow on motorways in a ramp metering problem). Through receiving a reward (either

positive or negative) after each execution of control actions, the agent can know how good its actions are.

The RL process usually follows the discrete-time mechanism, which means the learning process is discretised into a number of time steps. At each step, the agent interacts with its environment by taking a specific action and observing the change of environmental state with a relevant reward for this action. If this change leads to a good result, a positive reward will be received by the agent as an encouragement. If the change is undesirable, a negative reward will be received as a penalty. The objective of an agent is to get the maximum cumulative reward after executing a sequence of actions (Sutton and Barto 1998).

This section gives a general introduction about how RL works through the interaction between an agent and its external environment. To understand the detailed mechanism of RL, two kinds of problems, namely the Markov decision process (MDP) and dynamic programming (DP) should be known first. These two problems form the basis of RL.

3.2 The Markov Decision Process

Formally, RL is described as an MDP which can be represented by a 4-tuple $\{S, A, R(s,a,s'), P(s' \mid s,a)\}$ (Watkins 1989, Puterman 2009, Davarynejad et al. 2011).

- S is the state set (or state space) used to describe the external environment of an agent. In the discrete state case, a state set S is composed of a finite number of states. At each time step, a state $s \in S$ is observed by the agent to capture the environmental change.

- A is the action set, containing all executable actions for the agent. After observing the state at each time step, the agent chooses an action $a \in A$ to execute according to different action selection strategies.

- $R(s,a,s')$ is the reward function, which generates an immediate reward r ($r \in \mathbb{R}$) for the agent reaching state s' from state s after taking action a. r can be a positive or negative real number related to the goal (or objective) of an agent.

- $P(s'|s,a)$ is the state transition probability. For state pair $(s, s' \in S)$, $P(s'|s,a)$ represents the probability of reaching state s' after executing action a at state s ($\sum_{s' \in S} P(s'|s,a) = 1$).

Given the reward function and state transition probability, the expected reward $R(s,a)$ can be obtained by Equation (3.1), which is usually used to solve an MDP problem (Puterman 2009).

$$R(s,a) = \sum_{s' \in S} R(s,a,s') P(s'|s,a) \tag{3.1}$$

From the definition of MDP, it can be seen that the state transition probability $P(s'|s,a)$ to the next state s' is only determined by the current state s and action a, not all previous states and actions. This property is termed the "Markov property", based on which the value of the current state is sufficient for finding optimal actions for the next state (Puterman 2009). The Markov property determines whether a problem can be modelled as a Markov decision process, which will be mentioned again in Chapter 4, for defining RL elements in a ramp metering problem. Besides four basic elements, for solving an MDP problem, some other terminologies such as policies, returns and value functions should also be defined.

3.2.1 Policies and returns

As mentioned earlier, the aim of an agent is to maximise the cumulative reward by taking a sequence of actions. In a formal definition, the sequence of actions is determined by “policy” and the cumulative reward is expressed by “return”.

A policy π is defined as a mapping from observed states to executable actions ($\pi : S \to A$) (Watkins 1989). Therefore, the policy determines which action to take with observation of a specific state. If a policy with actions can lead to the maximum cumulative reward, this policy is defined as the optimal policy. Therefore, maximising the cumulative reward is equivalent to finding the optimal policy.

The return Re^t is used to aggregate a sequence of rewards (cumulative reward) after time step t. If the immediate reward generated by the reward function $R(s,a,s')$ at step t is r^t , then Re^t can be defined as a weighted sum of rewards of all the following steps (Sutton and Barto 1998).

$$Re^t = r^{t+1} + \gamma r^{t+2} + \cdots + \gamma^{(n)} r^{t+n+1} + \cdots = \sum_{n=0}^{\infty} \gamma^{(n)} r^{t+n+1} \tag{3.2}$$

The superscripts of Re and r are time step indices, while $\gamma^{(n)}$ means γ to the power n. γ ($\gamma \in [0,1]$) is the discount rate, which indicates that, with the increase of time step, the importance of its corresponding reward for calculating the return Re^t is decreasing.

3.2.2 Value functions

The return defined above only gives a sample of the cumulative reward, and a number of returns may exist in an MDP problem. Thus, to better express the cumulative reward, the expectation (denoted by $E\{\}$) of these returns

should be used. Usually, this expectation can be defined as two value functions, state-value function and action-value function (Sutton and Barto 1998).

Definition of value functions

Given a policy π, according to the Markov property, the expectation of returns can be formulated as a function of a state s, i.e. $V^{\pi}(s)$. This function is named the "state-value function" for policy π at state s (as given by Equation (3.3)).

$$V^{\pi}(s) = E\left\{Re^{t} \mid s^{t} = s, \pi\right\} = E\left\{\sum_{n=0}^{\infty} \gamma^{(n)} r^{t+n+1} \mid s^{t} = s, \pi\right\} \tag{3.3}$$

Besides the state s, if the action a is also one variable that can determine the cumulative reward, another similar function named the "action-value function" ($Q^{\pi}(s,a)$) can be defined by Equation (3.4), which is for policy π at state s after taking action a.

$$Q^{\pi}(s,a) = E\left\{Re^{t} \mid s^{t} = s, a^{t} = a, \pi\right\} = E\left\{\sum_{n=0}^{\infty} \gamma^{(n)} r^{t+n+1} \mid s^{t} = s, a^{t} = a, \pi\right\} \tag{3.4}$$

This function can also be called the Q function, and the value of the Q function is known as the Q value.

Bellman equation

One important feature of value functions is that they can be updated at each step, which provides a way for developing relevant algorithms to solve MDP problems iteratively. Take the state-value function for example, there exists a recursive relationship between the value of the current state and its possible next states, through which optimal values and policies can be updated recursively by proper algorithms. This relationship is called the

Bellman equation (Kaelbling et al. 1996) which can be derived from Equation (3.3) and expressed as follows:

$$\begin{aligned} V^{\pi}(s) &= E\left\{\sum_{n=0}^{\infty}\gamma^{(n)}r^{t+n+1} \mid s^{t}=s,\pi\right\} \\ &= E\left\{r^{t+1}+\gamma\sum_{n=0}^{\infty}\gamma^{(n)}r^{t+n+2}\,\middle|\,s^{t}=s,\pi\right\} \\ &= \sum_{s'\in S}P(s'\mid s,a)\left[r^{t+1}+\gamma E\left\{\sum_{n=0}^{\infty}\gamma^{(n)}r^{t+n+2}\,\middle|\,s^{t+1}=s',\pi\right\}\right] \\ &= R(s,a)+\gamma\sum_{s'\in S}P(s'\mid s,a)V^{\pi}(s') \end{aligned} \tag{3.5}$$

Based on the Bellman equation, the optimal value $V^{*}(s)$ $(V^{*}(s)=\max_{\pi}V^{\pi}(s))$ of state s can be obtained by the equation below:

$$V^{*}(s)=\max_{a\in A}\left[R(s,a)+\gamma\sum_{s'\in S}P(s'\mid s,a)V^{*}(s')\right] \tag{3.6}$$

Thus, the optimal policy $\pi^{*}(s)$ can be defined as the policy that can lead to the maximum value of state s.

$$\pi^{*}(s)=\arg\max_{a\in A}\left[R(s,a)+\gamma\sum_{s'\in S}P(s'\mid s,a)V^{*}(s')\right] \tag{3.7}$$

Similarly, for the action-value function, the optimal value of the state-action pair (s,a) can be expressed by:

$$Q^{*}(s,a)=R(s,a)+\gamma\sum_{s'\in S}P(s'\mid s,a)\max_{a'}Q^{*}(s',a') \tag{3.8}$$

Based on the Bellman equation, some algorithms such as DP algorithms have been developed to solve MDP problems. The next section will introduce these algorithms.

3.3 Dynamic Programming

Given a model of state transition probabilities and reward function, DP provides a possible way to solve an MDP problem. In this section, two basic algorithms of DP, namely policy iteration and value iteration are introduced.

3.3.1 Policy iteration

The policy iteration algorithm can be found in Algorithm 3.1 (Kaelbling et al. 1996). The basic idea of policy iteration is that the algorithm begins with an arbitrarily selected policy and then improves it iteratively. Specifically, two core steps (lines 4 and 5 of Algorithm 3.1) of the algorithm are repeated to get the optimal policy.

Algorithm 3.1: Policy iteration

1. initialise a policy π' arbitrarily
2. **repeat**
3. $\pi \leftarrow \pi'$
4. calculate the value for π
   by solving the linear equations:
   $$V^{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} P(s' \mid s, \pi(s)) V^{\pi}(s')$$
5. improve the policy for each $s \in S$
   $$\pi'(s) = \arg\max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} P(s' \mid s, a) V^{\pi}(s') \right]$$
6. **until** π' is the same as π

(1) The first core step (line 4) is named the policy evaluation. $|S|$ Linear equations presented by $V^{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} P(s' \mid s, \pi(s)) V^{\pi}(s')$ (one for each state $s \in S$) are solved in this step to get the expected value $V^{\pi}(s)$ under policy π for each state s. Here, $\pi(s) = a$.

(2) The second core step (line 5) is termed the policy improvement. Any new policy π' with the maximum $V^{\pi}(s)$ is used to replace the current policy π .

This algorithm ends when no policies are better than the current one (when $\pi = \pi'$). In this way, the optimal policy can be found after a number of iterations.

3.3.2 Value iteration

The value iteration algorithm can be found in Algorithm 3.2 (Watkins 1989).

Algorithm 3.2: Value iteration

1. initialise $V^0(s)$ arbitrarily, $i = 0$
2. **repeat**
3. $i \leftarrow i+1$
4. **for** each $s \in S$ **do**
5. $V^i(s) \leftarrow \max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} P(s' \mid s,a) V^{i-1}(s') \right]$
6. **until** the differences between $V^i(s)$ and $V^{i-1}(s)$ are small enough for all s

Because the policy π is not known, the value function $V^i(s)$ instead of $V^{\pi}(s)$ is used here to denote the value updated in the i th iteration. Unlike policy iteration, this algorithm starts with an arbitrary value $V^0(s)$ for each $s \in S$. After that, it chooses an action $a \in A$ iteratively to improve the values (that is why it is called value iteration). For the i th iteration, the value of each state $V^i(s)$ is computed according to the value of each state $V^{i-1}(s)$ from the last iteration $i-1$. The action a that can lead to the maximum $V^i(s)$ for each state is recorded in the policy. In this way, through continuously

improving $V^i(s)$ for a number of iterations, the algorithm can learn the policy corresponding to the maximum $V^i(s)$ for each state.

In real applications, when the differences between $V^i(s)$ and $V^{i-1}(s)$ for all states are smaller than a predefined value (usually a small positive number), the algorithm will end.

Two basic DP methods (policy iteration and value iteration) used to solve MDP problems have been introduced in this section. One important precondition of DP is knowing of a model related to the transition probability $P(s'|s,a)$ and reward function $R(s,a,s')$. However, it is very difficult for many practical applications (such as ramp metering problems) to get this model in advance. Under such circumstances, the agent needs to interact with its external environment directly and learn the optimal policy without the help of models. This concept of "learning without models" forms the basis of some important RL-based algorithms. The next section will introduce the mechanisms of these algorithms.

3.4 Temporal Difference Learning

Without a model of transition probabilities and reward functions, an alternative way of solving MDP problems is the temporal difference (TD) learning. TD learning is the core idea of RL that can help the agent learn how to find suitable solutions by directly interacting with its environment. TD (or TD (λ)) contains a group of algorithms extended from the basic DP. TD(λ) learning is a model-free method which tries an action at a state, and estimates its related value according to the immediately received reward and the value of the next state without knowing all state transition probabilities and rewards (Kaelbling et al. 1996). λ here is a parameter related to the so-called eligibility trace of each state, which determines how many rewards are

considered in the return (cumulative reward) and which states are eligible for updating the state values ($V^t(s)$) (Sutton and Barto 1998) . Similar to the value iteration, $V^t(s)$ here is the value updated at time step t.

3.4.1 SARSA and Q-learning

A particular case of TD (λ) with λ = 0 is a TD(0) algorithm where only one immediate reward is considered in the return. Because of its simplicity, this algorithm has been widely used in practice, and this study will focus on TD(0) only.

Through the updating rule (3.9), at each time step t, the value $V^t(s^{t-1})$ of the last visited state s^{t-1} is updated to approach $r^t + \gamma V^{t-1}(s^t)$ which is the sum of the immediate reward r^t and the discounted value of the current state $\gamma V^{t-1}(s^t)$. Because r^t is the real received reward, through each updating, $r^t + \gamma V^{t-1}(s^t)$ should get closer to the real value of this state $V(s^{t-1})$. Thus, the optimal value of each state can be obtained through recursively calling the updating rule (3.9).

$$V^t(s^{t-1}) = V^{t-1}(s^{t-1}) + \alpha\left[r^t + \gamma V^{t-1}(s^t) - V^{t-1}(s^{t-1})\right] \quad (3.9)$$

where, α is the learning rate or step-size parameter that is used to determine how fast values of states can be updated approaching their maximum values (Even-Dar and Mansour 2004). Typically, α is a small positive fraction value within the range between 0 and 1.

SARSA learning

Equation (3.9) shows how the state-value function is updated according to the basic idea of TD(0) learning. If this equation is extended to update the action-value function (Q value), a new TD(0) algorithm named “SARSA” (state-action-reward-state-action) with the updating rule (3.10) can be derived from (3.9) (Sutton and Barto 1998).

$$Q^t(s^{t-1},a^{t-1}) = Q^{t-1}(s^{t-1},a^{t-1}) + \alpha\left[r^t + \gamma Q^{t-1}(s^t,a^t) - Q^{t-1}(s^{t-1},a^{t-1})\right] \quad (3.10)$$

SARSA is an on-policy TD, because the Q value update is directly dependent on the action executed according to some sort of action selection strategy, which means $Q^t(s^{t-1},a^{t-1})$ is calculated according to $Q^{t-1}(s^t,a^t)$.

Q-learning

For an off-policy method, $\max_{a^t} Q^{t-1}(s^t,a^t)$ instead of $Q^{t-1}(s^t,a^t)$ is used to update the Q value. In this way, the greedy action corresponding to the optimal Q value is selected to update Q values, which is independent from the real executed action. This kind of learning method forms another TD(0) algorithm called “Q-learning”. The updating rule for Q-learning is given below (Sutton and Barto 1998):

$$Q^t(s^{t-1},a^{t-1}) = Q^{t-1}(s^{t-1},a^{t-1}) + \alpha\left[r^t + \gamma \max_{a^t} Q^{t-1}(s^t,a^t) - Q^{t-1}(s^{t-1},a^{t-1})\right] \quad (3.11)$$

With suitable parameters, for the same problem, both SARSA and Q-learning can converge to the optimal policy, while Q-learning has an earlier convergence in many cases (Sutton and Barto 1998). In practical applications, Q-learning is more popular than SARSA learning.

3.4.2 Action selection strategies

For a RL-based agent, exploitation and exploration are two basic behaviours (Kaelbling et al. 1996). Exploitation means the agent always takes the greedy action that can obtain the maximum cumulative reward (such as the Q value in Q-learning) according to the existing experience. Exploration is the behaviour when the agent tries non-greedy actions with smaller cumulative reward. These two behaviours are essential for the continuous learning of RL. Exploration can help the agent discover new actions that may be better than the greedy actions found previously (because of the new

information captured). In the meanwhile, exploitation can keep the agent from being interrupted too often by the exploration.

In order to balance these two behaviours, some action selection strategies are developed for selecting suitable executed actions, among which the so-called "ε-greedy" is the most commonly used strategy. Specifically, this strategy takes a random non-greedy action ($a^t \neq a^t_{greedy}$) with probability ε and chooses the greedy action ($a^t = a^t_{greedy}$) with probability $1-\varepsilon$ at each state s^t (as shown in Equation (3.12)). The greedy action a^t_{greedy} at state s^t is the action corresponding to the maximum Q value at this state.

$$p(a^t \mid s^t) = \begin{cases} \varepsilon, & \text{if } a^t \neq a^t_{greedy}, a^t_{greedy} = \arg\max_{a^t}(Q^{t-1}(s^t, a^t)) \\ 1-\varepsilon, & \text{otherwise} \end{cases} \tag{3.12}$$

When ε ($\varepsilon \in [0,1]$) is larger, the agent will be more adventurous and always try to explore the unknown actions. This kind of exploration may be good, and better actions may be found much faster than using a conservative strategy. However, it may also interrupt the learning process by trying worse actions too often.

3.5 Multi-objective Reinforcement Learning

So far the basic mechanism and related algorithms of RL with one control objective has been introduced. In many practical applications, decisions should be taken on the basis of the trade-off between multiple objectives, which are usually formulated as multi-objective optimisation problems (MOO). The aim of MOO is to achieve one or more acceptable compromises of all desired objectives (Ngatchou et al. 2005). These acceptable compromises can form a so-called Pareto front with non-dominated solutions.

In order to solve MOO problems, the basic reinforcement learning has been extended to multi-objective reinforcement learning (MORL) with a number of emerging algorithms (Vamplew et al. 2011). In this section, some relevant MORL algorithms will be briefly introduced. Generally, the MORL algorithms can be classified into two main categories namely single policy and multi policy algorithms according to the number of policies needed by a problem domain (Vamplew et al. 2011).

3.5.1 Single-policy algorithms

In single-policy scenarios, only one policy is required according to some predefined criteria. Most algorithms of MORL are focused on single policy learning (Roijers et al. 2013).

W-learning is one extension of single-objective Q-learning that uses multiple agents to learn multiple Q values, each of which corresponded to an objective (Humphrys 1995). The final decision was made based on negotiation between all agents involved and the action proposed by the “winning” agent was selected as the executed action. This “winner takes all” method can guarantee that the action selected is optimal for at least one objective. In recent years, this method was extended to distributed scenarios and applied to solve urban traffic control problems (Dusparic and Cahill 2009). Although this method is efficient for many problems, its drawback is also obvious. As mentioned by (Roijers et al. 2013) this method may fail to find the solution (or policy) that should be a compromise of different objectives.

Without using multiple agents, Gábor et al. (1998) proposed a general framework for solving MORL problems. One objective can be maximised by making some constraints to other objectives. Through setting preferences for different objectives, a good compromise between these objectives can be

obtained. In some cases, the information about preferences on different objectives cannot be directly obtained in advance. Under such circumstances, fuzzy logic was applied in (Yun et al. 2010) to quantify preferences based on the relative importance between different pairs of objectives.

Obviously, specific knowledge about the problem domain is required to set constraints and quantify preferences for different objectives. This knowledge is usually difficult to obtain in advance. Another way to solve this problem is by using the linear scalarisation method (Vamplew et al. 2011). Each objective in the problem domain can be assigned a weight value, and the weighted sum of all objectives can form a new objective that should be maximised (or minimised). The relative importance between different objectives can be regulated by setting different weight values. By running the algorithm several times under different weight settings, the user can select the acceptable solution from a set of generated solutions. Thus, no pre-existing knowledge is required.

3.5.2 Multi-policy algorithms

For multi-policy cases, a number of possible policies that can generate the Pareto front are required. In many practical cases, the objective of an algorithm is to find one or more suitable solutions to solve the problem encountered and there is no need to find all solutions forming the theoretical Pareto front. Moreover, generating a number of policies requires high computational demand that drastically increases the cost in time. Thus, not too much work has been done on multi-policy algorithms, especially when practical applications are considered. One example related to multiple policies can be found in (Barrett and Narayanan 2008) where optimal policies related to different preferences can be learnt in parallel. As mentioned in (Vamplew et al. 2011), the linear scalarisation method can also

be used to find multiple policies by regulating the weight values one at a time and running the algorithm several times. However, this method may not be able to find all possible solutions to form the Pareto front (Vamplew et al. 2011).

3.5.3 Linear scalarised Q-learning

As a practical application of RL, this thesis focuses on finding suitable solutions to balance different objectives for ramp metering, not a theoretical Pareto front. The linear scalarised algorithm that does not need pre-existing knowledge will be of interest here. Although this method may not be able to find all solutions to form a Pareto front, several possible options of balancing different control objectives are enough for a practical control problem such as ramp metering. The most acceptable one among these solutions can be selected by operators according to different requirements about the importance of different objectives (such as improving traffic efficiency and maintaining user equity).

One commonly used way for linear scalarisation in RL is to extend the single-objective Q-learning to linear scalarised Q-learning. Compared with single-objective case, two differences arise in linear scalarised Q-learning (Vamplew et al. 2011): (1) multiple rewards are received at each time step for different objectives, (2) action selection is based on the scalarised Q value related to all objectives involved. Thus, how to scalarise different objectives is one important issue in scalarised Q-learning. An effective method mentioned in (Van Moffaert et al. 2013) is introduced here to linearly scalarise Q values. The scalarised Q value can be expressed by the following equation:

$$SQ^t(s,a) = \sum_{j=1}^{N_j} \left[\delta_j \cdot Q_j^t(s,a) \right] \tag{3.13}$$

where, $SQ^t(s,a)$ is the scalarised Q value for state-action pair (s,a), $Q_j^t(s,a)$ is the Q value for objective j, N_j is the number of objectives considered, δ_j is the weight value for $Q_j^t(s,a)$ and $\sum_{j=1}^{N_j} \delta_j = 1$. The Q value for each objective should be updated following a new updating rule shown below.

$$Q_j^{\,t}(s^{t-1}, a^{t-1}) = Q_j^{\,t-1}(s^{t-1}, a^{t-1}) + \alpha \left[r^t + \gamma \cdot Q^{t-1}(s^t, a^t_{greedy}) - Q_j^{\,t-1}(s^{t-1}, a^{t-1}) \right] \tag{3.14}$$

Instead of $a^t_{greedy} = \arg\max_{a^t}(Q^{t-1}(s^t, a^t))$, the greedy action in the linear scalarised Q-learning is selected according to $a^t_{greedy} = \arg\max_{a^t}(SQ^{t-1}(s^t, a^t))$.

3.6 Summary and Discussion

This chapter introduced the basic knowledge of RL and one of its extensions to solve multi-objective problems.

As the core algorithm of RL, TD learning was developed on the basis of the basic policy iteration algorithm. TD learning contains a series of algorithms that can solve MDP problems adaptively without the help of models. Q-learning is one off-policy TD algorithm which has been widely used to solve practical problems, especially in the ramp metering domain, because this algorithm is efficient and easy to implement.

However, the basic Q-learning lacks the ability to deal with multiple objectives. To overcome this limitation, one extension of Q-learning, linear scalarised Q-learning was proposed to solve multi-objective problems. One advantage of this algorithm is that it does not need pre-existing knowledge of the problem domain, and a set of solutions can be generated by it through properly regulating related parameters.

Because of the simplicity of implementation, the Q-learning and its extension, linear scalarised Q-learning, will be considered by this research to develop related control algorithms in a ramp agent system. The detailed design process of this ramp agent system and two control algorithms using RL (Q-learning and linear scalarised Q-learning) will be introduced in the following chapter.

CHAPTER 4 RAMP AGENT SYSTEM

Chapter 3 gave a detailed introduction of RL including its structure, mechanism and algorithms. The basic agent-environment architecture of RL introduced in Section 3.1 will be extended to deal with ramp metering problems in this chapter. Based on the RL mechanism, a ramp agent system (RAS) that can learn how to control motorway traffic via ramp metering will be developed. This chapter gives a systematic description of the RAS design regarding its structure, elements, modules and algorithms. The sections of this chapter are organised as follows.

Section 4.1 firstly presents the basic architecture of agent-environment interaction in a ramp metering problem including RAS and the controlled motorway. Then, how the controlled motorway can be simulated by a traffic flow model is discussed in Section 4.2, which provides traffic related information to RAS. Based on this information, the detailed definition and design of RAS are given in Section 4.3. After that, Section 4.4 presents two control algorithms for RAS, which can be used to deal with both single- and multi-objective problems of ramp metering. Finally, a discussion of this chapter is given in Section 4.5.

4.1 Ramp Agent and Environment

4.1.1 General architecture

It has been introduced in Chapter 2 that the main work of a ramp metering strategy especially the traffic-responsive strategy is to convert the traffic information collected from motorways to suitable metering rates. In this study, the controller that can implement a RL-based strategy is the ramp agent. A system that is composed of a group of agents is known as a ramp agent

system (RAS). The whole control process [5] is conducted through the interaction between each ramp agent and its controlled motorway.

Based on the motorway layout and detector locations, the controlled motorway is usually divided into several segments. For ease of modelling and expression, each controlled segment contains only one metered on-ramp (Becerril-Arreola and Aghdam 2007). Following this partition method, an example for practical application of one ramp agent is presented in Figure 4.1. This architecture is extended from the basic agent-environment interaction shown in Figure 3.1 of Section 3.1.

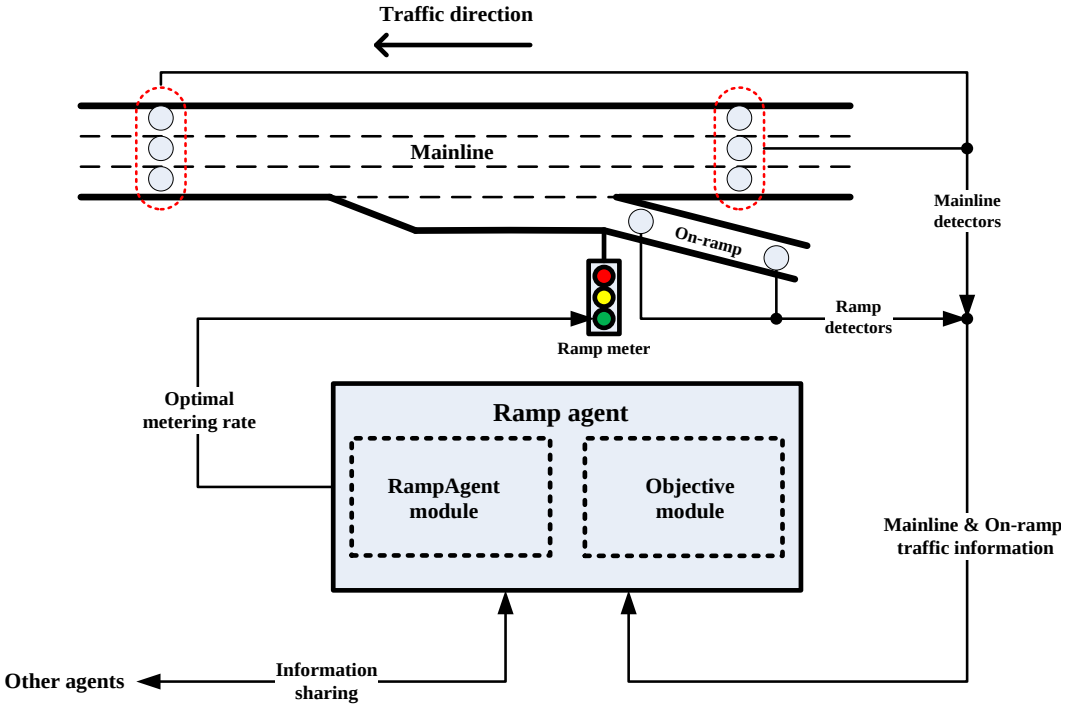


Figure 4.1: Ramp agent and motorway interaction

[5] In this thesis, a “control cycle” or “control process” of a ramp agent is also known as the “learning process”. In the following sections of this thesis, two terms “control” and “learning” may be used alternatively according to different contexts.

The environment in this case refers to the controlled motorway segment, while the agent is extended to a ramp agent that controls this segment. At the beginning of each control cycle, the ramp agent obtains traffic information (e.g. density, flow and speed) from detectors located at the motorway mainline and on-ramp. Then, it generates optimal metering rates for the related ramp meter. If necessary, these metering rates can be further converted to specific signal timings for field applications.

The ramp agent is developed on the basis of RL mechanism. Two main modules, the ***RampAgent*** module and the ***Objective*** module are included in the agent structure. The ***RampAgent*** module contains sub-modules that can convert the raw traffic information into states and actions. To deal with different control objectives, a specific ***Objective*** module responsible for calculating rewards and updating related Q values is maintained. The learning process of a ramp agent can be accomplished by these two modules and their corresponding sub-modules. The more detailed description of these sub-modules will be presented in Section 4.3.

4.1.2 Working mechanism of RAS

Before the detailed introduction of a specific ramp agent, the problem of how the agents work with each other to form a system is discussed here.

In this study, two main control objectives related to ramp metering, i.e. improving traffic efficiency and maintaining user equity are included in the design of RAS. According to the number of objectives considered, RAS has two modes: a single-objective mode and a multi-objective mode.

In the single-objective mode, traffic efficiency is the only objective considered. When this mode is triggered, each ramp agent in RAS only captures the traffic information from its own controlled motorway segment. The objective of each ramp agent is to improve traffic efficiency within its

own control range. Thus, under the single-objective mode, RAS can be considered as a local control strategy, which only focuses on the local information.

In the multi-objective mode, user equity is involved as an additional control objective. Since each controlled segment only contains one on-ramp, the information from other motorway segments is essential to maintaining. In this situation, ramp agents in RAS need to share information with each other and work together to achieve this objective. Therefore, RAS becomes a coordinated strategy when the multi-objective mode is triggered.

Through these two modes, ramp agents in RAS can work independently to pursue their local objectives, or work together for a common goal. The details about what information should be captured from the local motorway segment and what information is required from other agents will be discussed in Section 4.3.

4.2 Controlled Motorway: ACTM

Usually, the newly developed control strategy cannot be directly tested in a real motorway network. Traffic flow models that can simulate the real traffic operation are commonly used as tools to evaluate traffic control strategies in the traffic engineering domain. This study also uses a traffic flow model to evaluate the proposed RAS. For the practical application, as suggested by (Jacob and Abdulhai 2010, El-Tantawy et al. 2013), the ramp agent can learn the optimal control strategy from a simulation model first, and then use that strategy to control the real traffic.

A macroscopic traffic flow model named the asymmetric cell transmission model (ACTM) is selected for this study, because this model is computationally efficient and has shown its effectiveness on evaluating ramp metering strategies in some recent studies (Gomes and Horowitz 2006,

Haddad et al. 2013, Sun and Horowitz 2006). ACTM was developed specially for simulating traffic flows on motorways and derived from the famous cell transmission model (CTM). This section will briefly introduce the development of CTM and how ACTM is derived.

4.2.1 Cell transmission model

LWR model

CTM is a very commonly used traffic flow model for simulating traffic dynamics on both urban and interurban road networks (Daganzo 1994, Daganzo 1995). This model is a finite difference approximation of the Lighthill-Whitham-Richards (LWR) model, which was the first macroscopic traffic flow model inspired from hydrodynamics (Lighthill and Whitham 1955, Richards 1956). This model assumed that the traffic flow satisfies the principle of mass conservation (or vehicle conservation for traffic) and gave the equation below:

$$\frac{\partial \rho(y,z)}{\partial z} + \frac{\partial q(y,z)}{\partial y} = 0 \tag{4.1}$$

where, $\rho(y,z)$ denotes the density that is a function of location y and time z, $q(y,z)$ is the flow related to location y and time z. By introducing a relationship between the traffic flow and density under the equilibrium flow situation shown in Equation (4.2), the LWR model can be written as a solvable partial differential equation given by (4.3).

$$q(y,z) = q_e(\rho(y,z)) \tag{4.2}$$

$$\frac{\partial \rho(y,z)}{\partial z} + \frac{dq_e(\rho(y,z))}{d\rho} \cdot \frac{\partial \rho(y,z)}{\partial y} = 0 \tag{4.3}$$

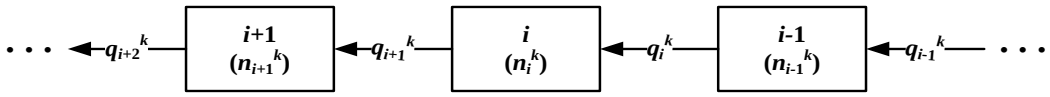


Figure 4.2: Cell connection for CTM

CTM

Considering the principle of vehicle conservation, CTM tried to discretise this continuum LWR model and solved it with a finite difference approximation method (Daganzo 1994, Daganzo 1995). Specifically, in CTM, the road studied is divided into a group of short segments called cells. As an example shown in Figure 4.2, each cell is assigned an index i and has a length l_i. The time period for simulation is also divided into a sequence of intervals and each interval has a duration of T_s with a time step index k. If v_i is the free flow speed, the cell length l_i should be set according to:

$$\min_i l_i \geq v_i \cdot T_s \tag{4.4}$$

This is called Courant–Friedrichs–Lewy (CFL) condition. Short cell length that does not satisfy the CFL condition will lead to the invalidation of vehicle conservation given by Equation (4.2). If the cell length is too long, model accuracy may not be guaranteed. Thus, the cell length of CTM is usually set from 100 to 1000 metres. With suitable settings of cell lengths and time intervals, the discrete vehicle conservation of CTM can be expressed by:

$$n_i^{\ k+1} = n_i^{\ k} + q_i^{\ k} - q_{i+1}^{\ \ k} \tag{4.5}$$

where, $n_i^{\ k}$ and $n_i^{\ k+1}$ are the number of vehicles on the cell i at time step k and $k+1$. $q_i^{\ k}$ and $q_{i+1}^{\ \ k}$ are inflows of cell i and $i+1$ at time step k. $q_i^{\ k}$ is determined by the following equation:

$$q_i^{\ k} = \min\left\{q_{S,i-1}^{\ \ k}; q_{R,i}^{\ \ k}\right\} \tag{4.6}$$

where, $q_{S,i-1}{}^{k}$ represents the number of vehicles that can be sent by cell $i-1$ at time step k during a time period T_s, while $q_{R,i}{}^{k}$ is the number of vehicles that can be received by cell i at time step k during a time period T_s.

These two values can be calculated by Equations (4.7) and (4.8) according to the triangular or trapezoidal fundamental diagram shown in Figure 4.3.

$$q_{S,i-1}{}^{k} = \min\left\{ n_{i-1}{}^{k}; q_{cap,i-1} \right\} \tag{4.7}$$

$$q_{R,i}{}^{k} = \min\left\{ q_{cap,i}; (w_i / v_i) \cdot (n_i^{\max} - n_i{}^{k}) \right\} \tag{4.8}$$

where, w_i is the congestion wave speed, ρ_i is the density of cell i, $\rho_{jam,i}$ is the jam density of cell i, $q_{cap,i-1}$ is the maximum number of vehicles that can be sent by $i-1$ during T_s, $q_{cap,i}$ is the maximum number of vehicles that can be received by i during T_s, $n_i = \rho_i \cdot l_i$ and $n_i^{\max} = \rho_{jam,i} \cdot l_i$. In the original CTM shown in (Daganzo 1994, Daganzo 1995), it was assumed that both T_s and v_i were unit values(T_s = 1 and v_i = 1), and l_i is chosen as $l_i = v_i \cdot T_s$ =1.

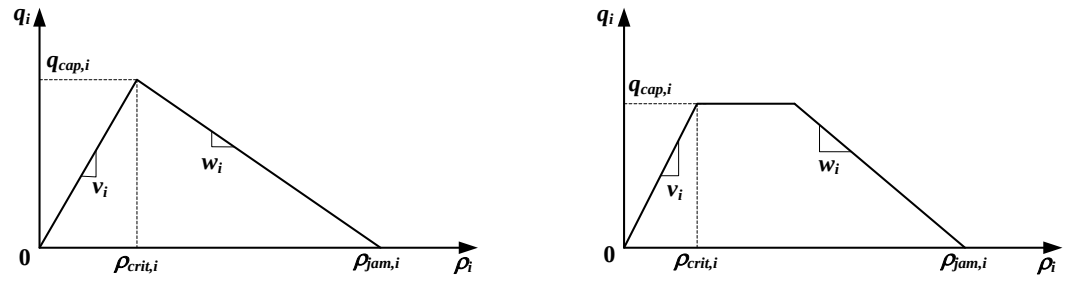


Figure 4.3: Triangular and trapezoidal fundamental diagrams

Extension of CTM

Considering more general road networks, Daganzo (Daganzo 1995) extended the basic CTM to simulate more complicated traffic flow dynamics such as merging and diverging conditions.

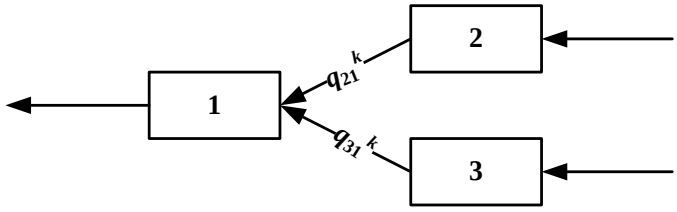


Figure 4.4: Merging model of CTM

For a merging situation, as shown in Figure 4.4, traffic flows coming from two different upstream cells (cell 2 and 3) need to merge into one cell (cell 1). Here, $q_{21}^{\ k}$ is used to represent the traffic flow coming from cell 2 to cell 1, and $q_{31}^{\ k}$ is the traffic flow from cell 3 to cell 1. These two flows can be obtained through the following two equations:

$$q_{21}^{\ k} = \mathrm{mid}\left\{q_{S,2}^{\ k};\ q_{R,1}^{\ k} - q_{S,2}^{\ k};\ p_{21}\cdot q_{R,1}^{\ k}\right\} \tag{4.9}$$

$$q_{31}^{\ k} = \mathrm{mid}\left\{q_{S,3}^{\ k};\ q_{R,1}^{\ k} - q_{S,3}^{\ k};\ p_{31}\cdot q_{R,1}^{\ k}\right\} \tag{4.10}$$

In above equations, $\mathrm{mid}\{\}$ function is used to get the middle value of three involved arguments. p_{21} and p_{31} are used to assign different priorities for traffic flows from cell 2 and cell 3. These two parameters satisfy that $p_{21}, p_{31} \in [0,1]$ and $p_{21} + p_{31} = 1$.

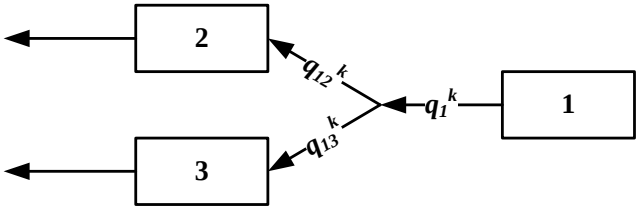


Figure 4.5: Diverging model of CTM

Figure 4.5 shows the diverging traffic condition. The flow from cell 1 is divided into two parts with one part entering cell 2 and the other going to cell 3. $q_{12}^{\ k}$ is the flow from cell 1 to cell 2, while $q_{13}^{\ k}$ denotes the flow from cell 1 to cell 3. If $q_1^{\ k} = q_{12}^{\ k} + q_{13}^{\ k}$, then the flows entering cells 2 and 3 can be computed with:

$$q_1^{\ k} = \min\left\{q_{S,1}^{\ k}; q_{R,2}^{\ k} / \beta_{12}; q_{R,3}^{\ k} / \beta_{13}\right\} \tag{4.11}$$

$$\begin{cases} q_{12}^{\ k} = \beta_{12} \cdot q_1^{\ k} \\ q_{13}^{\ k} = \beta_{13} \cdot q_1^{\ k} \end{cases} \tag{4.12}$$

In the above equations, β_{12} and β_{13} are split ratios determining portions of the flow coming from cell 1 ($q_1^{\ k}$) that enters cells 2 and 3 at time step k, and $\beta_{12} + \beta_{13} = 1$.

4.2.2 Asymmetric cell transmission model

The CTM and its extension introduced above can be used to simulate traffic dynamics in a more general case. For motorway traffic, it shows some limitations regarding the use of ramp metering strategies. To overcome these limitations, the asymmetric cell transmission model (ACTM) was developed by (Gomes and Horowitz 2003, Gomes and Horowitz 2006). This section will introduce this model.

Limitations of CTM

The limitations of CTM and what improvements have been made by ACTM are summarised below.

Using a symmetric merge model given by Equations (4.9) and (4.10), CTM does not distinguish traffic flows from different merging cells (cells 2 and 3), and models these flows without difference. On the other hand, ACTM separates the on-ramp follow from the mainline traffic flow model, and

clearly distinguishes these two flows when considering ramp metering. With this modification, the on-ramp traffic flow can be easily linked with ramp metering strategies, which provides convenience for testing different strategies.

The other limitation of the original CTM is that it cannot mimic the capacity drop phenomenon, which is one of the important reasons for applying ramp metering to improve traffic efficiency (as discussed in Section 2.2.1). ACTM can reproduce this phenomenon by applying a discontinuous variation. In this way, ACTM is more effective on testing ramp metering strategies when the traffic efficiency is considered as an objective.

Roughly speaking, CTM is more general, while ACTM was developed specifically for motorway ramp metering control.

Definition of ACTM

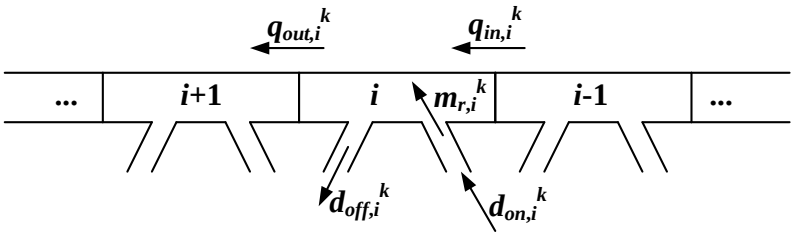


Figure 4.6: A typical motorway network for ACTM

Similar to the basic CTM, the motorway studied for ACTM should also be divided into short cells. Each cell may only contain the mainline, and may also be linked with on- or/and off-ramps. To simplify the expression in this study, the cell with one on- or off-ramps is named the “on-ramp cell” or the “off-ramp cell”, the cell with a pair of on- and off-ramps is named the “on-off cell”, and the cell without any ramps is named the “normal cell”. A typical cell (cell i) with one on-ramp and one off-ramp is shown in Figure 4.6, according to which the ACTM can be written as follows.

- Mainline flows

$$q_{out,i}^{k} = \min\{(1-\beta_i)\cdot\frac{v_i}{l_i}\cdot(n_{main,i}^{k}+\theta_i\cdot m_{r,i}^{k}\cdot T_s); \\ \frac{w_{i+1}}{l_{i+1}}\cdot(n_{main,i+1}^{\max}-n_{main,i+1}^{k}-\theta_{i+1}\cdot m_{r,i+1}^{k}\cdot T_s); q_{cap,i}; \frac{1-\beta_i}{\beta_i}\cdot d_{off,i}^{\max}\} \tag{4.13}$$

- On-ramp flows

$$m_{r,i}^{k} = \begin{cases} \min\{(n_{on,i}^{k}+d_{on,i}^{k}\cdot T_s)/T_s; \eta_i\cdot(n_{main,i}^{\max}-n_{main,i}^{k})/T_s; c_i^{k}/T_s\}, \\ \quad \text{if } i \text{ is metered on-ramp cell} \\ \min\{(n_{on,i}^{k}+d_{on,i}^{k}\cdot T_s)/T_s; \eta_i\cdot(n_{main,i}^{\max}-n_{main,i}^{k})/T_s\}, \\ \quad \text{if } i \text{ is unmetered on-ramp cell} \\ 0, \quad \text{otherwise} \end{cases} \tag{4.14}$$

- Mainline conservation

$$n_{main,i}^{k+1} = n_{main,i}^{k} + T_s\cdot(q_{in,i}^{k}+m_{r,i}^{k}-q_{out,i}^{k}/(1-\beta_i)) \tag{4.15}$$

- On-ramp conservation

$$n_{on,i}^{k+1} = n_{on,i}^{k} + T_s\cdot(d_{on,i}^{k}-m_{r,i}^{k}) \tag{4.16}$$

For ease of application, a more general expression by considering different cell lengths (l_i) and time intervals (T_s) is used here. If l_i and T_s are removed from above equations and parameters v_i, w_i, $n_{on,i}^{k}$ and $n_{main,i}^{k}$ are replaced by their normalised counterparts $v_{nor,i}$, $w_{nor,i}$, $n_{onnor,i}^{k}$ and $n_{mainnor,i}^{k}$, the same expression shown in (Gomes and Horowitz 2006) can be obtained. Here, $v_{nor,i} = (v_i\cdot T_s)/l_i$, $w_{nor,i} = (w_i\cdot T_s)/l_i$, $n_{onnor,i}^{k} = n_{onnor,i}^{k}/T_s$ and $n_{mainnor,i}^{k} = n_{main,i}^{k}/T_s$.

In Equations (4.13) to (4.16), to distinguish the inflow and outflow of a specific cell, two different flows $q_{in,i}^{k}$ and $q_{out,i}^{k}$ are defined. Two parameters η_i (flow allocation parameter) and θ_i (flow blending parameter) are defined specially for ACTM. η_i indicates the influence of mainline traffic density on on-ramp flows, which is related to the merging behaviour of the on-ramp and

mainline flows. θ_i determines how much of the on-ramp flow should be added to the mainline outflow calculation before the mainline inflow is joined at each time step. This parameter is set according to the location of the on-ramp. If the on-ramp is close to the exit boundary of cell i, θ_i should be larger, as more on-ramp flow has entered the mainline before mainline inflow arrives. If the on-ramp is located at the entering boundary of cell i, on-ramp flow and mainline inflow can enter cell i together and θ_i should be set as 0 (Gomes and Horowitz 2006).

4.2.3 Discontinuous ACTM

The original ACTM introduced in Section 4.2.2 can also be called continuous ACTM. For reproducing the capacity drop phenomenon, a discontinuous version of ACTM was developed by the same authors (Gomes and Horowitz 2003). They divided the mainline flow Equation (4.13) into several discontinuous equations given by (4.17). The capacity drop phenomenon can be simulated by considering a queue discharge rate $q_{dis,i}$, which should be less than capacity $q_{cap,i}$. Here the capacity drop parameter λ is used to correlate $q_{dis,i}$ and $q_{cap,i}$, such that $q_{dis,i} = \lambda \cdot q_{cap,i}$. $\lambda \in (0,1]$ denotes the percentage of capacity left after capacity drop. In this way, different capacity drops can be considered by regulating λ.

The only difference between continuous and discontinuous ACTM is the equation for calculating mainline flows, and other equations from (4.14) to (4.16) are the same. The mainline flow of discontinuous ACTM is computed with:

$$q_{out,i}^k = \begin{cases} (1-\beta_i)\cdot\dfrac{v_i}{l_i}\cdot(n_{main,i}^k + \theta_i \cdot m_{r,i}^k \cdot T_s), & \text{if } \rho_i^k \le \rho_{crit,i} \text{ and } \rho_{i+1}^k \le \rho_{crit,i+1} \\ \min\{(1-\beta_i)\cdot\dfrac{v_i}{l_i}\cdot(n_{main,i}^k + \theta_i \cdot m_{r,i}^k \cdot T_s); & \\ \dfrac{w_{i+1}}{l_{i+1}}\cdot(n_{main,i+1}^{\max} - n_{main,i+1}^k - \theta_{i+1}\cdot m_{r,i+1}^k \cdot T_s)\}, & \text{if } \rho_i^k \le \rho_{crit,i} \text{ and } \rho_{i+1}^k > \rho_{crit,i+1} \\ q_{dis,i}, & \text{if } \rho_i^k > \rho_{crit,i} \text{ and } \rho_{i+1}^k \le \rho_{crit,i+1} \\ \dfrac{w_{i+1}}{l_{i+1}}\cdot(n_{main,i+1}^{\max} - n_{main,i+1}^k - \theta_{i+1}\cdot m_{r,i+1}^k \cdot T_s), & \text{if } \rho_i^k > \rho_{crit,i} \text{ and } \rho_{i+1}^k > \rho_{crit,i+1} \end{cases} \quad (4.17)$$

In the above equation, $\rho_i^k = n_{main,i}^k / l_i$ and $\rho_{i+1}^k = n_{main,i+1}^k / l_{i+1}$. As introduced in Section 2.1, when the mainline density exceeds the critical density ($\rho_i^k > \rho_{crit,i}$), congestion will occur on motorways. In this situation, capacity drop phenomenon may arise. From Equation (4.17), it can be seen that when cell i is congested and cell $i+1$ is not congested, the maximum outflow (capacity) of cell i becomes $q_{dis,i}$ ($q_{dis,i} = \lambda \cdot q_{cap,i}$). If λ equals 1, Equation (4.17) is exactly the same as Equation (4.13), because no capacity drop is considered. In the remainder of this thesis, this discontinuous ACTM will be used to simulate traffic flow dynamics.

4.2.4 Relationships between ACTM and RAS

To correlate ACTM and RAS, two relationships between them should be clear here. If ACTM is used to simulate motorway traffic, the controlled motorway is divided into a number of cells, and each ramp agent has its own controlled cell (the term “cell” can be used to replace the word “segment” in a motorway network). For ease of expression, the controlled cell will have the same index of its corresponding ramp agent. For example, if cell i is under control of agent I, all variables relating to cell i can be converted to their counterparts with index I. Thus, it can be obtained that: $n_{mian,i} = n_{main,I}$, $n_{on,i} = n_{on,I}$, $q_{in,i} = q_{in,I}$, $q_{out,i} = q_{out,I}$, $d_{on,i} = d_{on,I}$, $c_i = c_I$, $m_{on,i} = m_{on,I}$, $\eta_i = \eta_I$,

$\theta_i = \theta_I$, $\beta_i = \beta_I$. In the reminder of this chapter, these agent-related variables will be used to define elements of the ramp agent, such as state, action and reward.

Besides the agent index, the other relationship is about the control step and simulation step. In ACTM, T_s is the duration of each simulation interval between two simulation steps (t). If T_c is used to denote the control interval between two control steps (k), then, $T_c = N_{cs} \cdot T_s$, which means each control interval can contain N_{cs} simulation steps. One example with $N_{cs} = 4$ is shown in Figure 4.7.

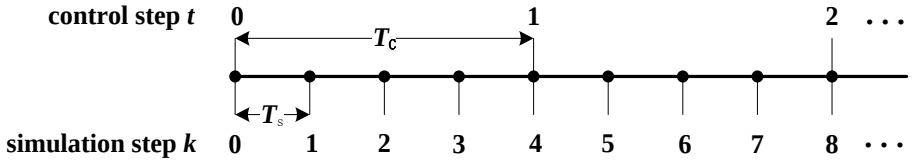


Figure 4.7: Control-simulation relationship

The reason for this setting is as follows. T_s is sometimes set as a very small value such as 10 or 15 seconds to guarantee the accuracy of simulation. However, it is not reasonable to change the metering rate within such a short time, because it may confuse drivers. For this reason, a suitable range should be set for T_c (usually 30 to 60 seconds as suggested by (Papageorgiou et al. 2007)) that may be a few times as long as T_s. This relationship is very important for RAS to convert control actions to suitable metering rates at each time step for ACTM. This relationship will be mentioned again in Section 4.3.5.

4.2.5 Summary

This section introduced a traffic flow model named ACTM regarding its origin and definition. The ACTM was developed specifically to deal with ramp

metering problems, because it can distinguish on-ramp flows and reproduce capacity drop phenomenon.

Besides the detailed description of ACTM, how to correlate ACTM and RAS was also discussed in this section. Two relationships in terms of the controlled cell and ramp agent, as well as the control step and simulation step were defined. Based on these two relationships, the detailed description of one ramp agent will be given in the next section.

4.3 Ramp Agent Design

Recall that, in Section 4.1, the basic structure of a ramp agent which contains two modules: the ***Rampagent*** module and the ***Objective*** module has been introduced. These two modules are detailed in this section with the definitions of related elements and sub-modules.

4.3.1 Element definition

As mentioned in Chapter 3, RL is used to solve MDP problems. But unlike the basic MDP, RL does not need to consider the state transition probability of the system. Therefore, for a specific application of RL, only three basic elements including state, action and reward should be defined. In this section, a general definition of these three elements in a ramp metering problem will be given.

Markov property

To formulate an effective RL problem, three elements of RL should be defined under the MDP framework which means they should satisfy the Markov property. The formal expression of the Markov property is given below (Puterman 2009):

$$p(s^{t+1} \mid s^0, a^0, ..., s^{t-1}, a^{t-1}, s^t, a^t) = p(s^{t+1} \mid s^t, a^t) \tag{4.18}$$

This equation means that given the state and action at time step t, the next state at step $t+1$ can be captured without knowing all the previous states and actions. In other words, the current state and action contain all the information required to determine the next state.

Reward

In a RL problem, reward is used to guide the agent to achieve its objectives. Therefore, the definition of reward should be derived from the control objectives. Let us get back to the ramp metering control problem introduced in Chapter 2. A general indicator measuring the performance of ramp metering strategies is TTS (Equation (2.2)) which is the total time spent on motorways. TTS is usually used to derive the control objectives and can be further divided into two parts: TTT and TWT (Kotsialos and Papageorgiou 2004a). The Equation (2.2) can be rewritten here using agent (I) and control step (t) indices:

$$TTS_I = T_c \cdot \sum_{t=0}^{N_t-1} (n_{main,I}^t + n_{on,I}^t) = \underbrace{T_c \cdot \sum_{t=0}^{N_t-1} n_{main,I}^t}_{TTT_I} + \underbrace{T_c \cdot \sum_{t=0}^{N_t-1} n_{on,I}^t}_{TWT_I} \tag{4.19}$$

where, TTS_I is the TTS of the cell controlled by agent I, which is composed of the total travel time on the mainline TTT_I and the total waiting time at the on-ramp TWT_I.

Through this modification, the definition of TTS becomes: $TTS = TTT + TWT$ which is similar to Equation (2.11) introduced in Section 2.2.2, i.e. $TWTT = WFTT + WRD$ ($TWTT$ is the total weighted travel time, $WFTT$ is the weighted mainline travel time, WRD is the weighted on-ramp delay). To obtain one common objective $TWTT$ that can balance the efficiency and equity, different weight values were assigned to $WFTT$ and WRD. In this study, instead of one common objective, two different objectives are defined

for the efficiency and equity respectively. In this way, a ramp agent can easily add or remove its control objectives according to different working modes.

In the single-objective mode, the only objective is to improve traffic efficiency, i.e. minimise TTS. As T_c is a fixed value, minimising TTS is equivalent to minimising the number of vehicles on both the motorway mainline and on-ramp $\sum_{t=0}^{N_t-1}(n_{main,I}^t + n_{on,I}^t)$. To minimise this value, the sum of numbers of vehicles on the mainline and on-ramp at each control step t is defined as a negative reward shown in Equation (4.20). This means that the more vehicles that are on the motorway, the more penalties will be received by the ramp agent.

$$r_{raw,I,1}^t = -(n_{main,I}^t + n_{non,I}^t) \tag{4.20}$$

In the multi-objective mode, except for efficiency improvement, an additional objective is to maintain user equity at different on-ramps. Following the definition of equity introduced in Section 2.2.2, perfect equity in this study means that users from different on-ramps can have the same total waiting time at on-ramps, i.e. the same TWT. Here, the standard deviation of TWT at different on-ramps is used to measure this equity, which can be expressed by:

$$SD(TWT) = \sqrt{\frac{\sum_{I=1}^{N_{on}}\left(TWT_I - \overline{TWT}\right)^2}{N_{on}}} = \sqrt{\frac{\sum_{I=1}^{N_{on}}\left(\sum_{t=0}^{N_t-1} n_{on,I}^t \cdot T_c - \overline{n}_{on} \cdot T_c\right)^2}{N_{on}}} \tag{4.21}$$

where, $\overline{TWT}$ is the average TWT of all on-ramps, $\overline{n}_{on}$ is the average cumulative on-ramp queue during the whole control period, which can be obtained by $\overline{n}_{on} = (\sum_{I=1}^{N_{on}} \sum_{t=0}^{N_t-1} n_{on,I}^t) / N_{on}$. To get the highest equity, $SD(TWT)$ should be minimised.

Since the reward is obtained at each time step, to derive the equity-related reward, $SD(TWT)$ at each time step should be known. Similar to $SD(TWT)$ of the whole control period, $SD^t(TWT)$ at step t can be calculated by:

$$\begin{aligned} SD^t(TWT) &= \sqrt{\frac{\sum_{I=1}^{N_{on}} \left(TWT_I^t - \overline{TWT}^t \right)^2}{N_{on}}} \\ &= \sqrt{\frac{\sum_{I=1}^{N_{on}} \left(n_{on,I}^t \cdot T_c - \overline{n}_{on}^t \cdot T_c \right)^2}{N_{on}}} \\ &= T_c \cdot \sqrt{\frac{\sum_{I=1}^{N_{on}} \left(n_{on,I}^t - \overline{n}_{on}^t \right)^2}{N_{on}}} \end{aligned} \tag{4.22}$$

where, $\overline{TWT}^t$ is the average TWT at step t, $\overline{n}_{on}^t$ is the average on-ramp queue at step t ($\overline{n}_{on}^t = (\sum_{I=1}^{N_{on}} n_{on,I}^t) / N_{on}$). Once again, T_c is a fixed value, minimising $SD^t(TWT)$ is equivalent to minimising $\sqrt{\sum_{I=1}^{N_{on}} \left(n_{on,I}^t - \overline{n}_{on}^t \right)^2 / N_{on}}$. Thus, the equity-related reward can be defined as a negative reward and expressed by:

$$r_{raw,I,2}^t = -\sqrt{\frac{\sum_{I=1}^{N_{on}} \left(n_{on,I}^t - \overline{n}_{on}^t \right)^2}{N_{on}}} \tag{4.23}$$

Both the efficiency-related reward $r_{raw,I,1}^t$ and equity-related reward $r_{raw,I,2}^t$ defined in this section are both raw rewards which should be normalised before the real application. The normalisation process will be introduced in Section 4.3.3.

State and action

From the definition of rewards, it can be seen that $n_{main,I}^t$ and $n_{on,I}^t$ contain the direct information relating to control objectives of a ramp agent. These

two variables will be used to derive the definitions of state and action. The evolution of $n^t_{main,I}$ and $n^t_{on,I}$ over time can be obtained from the vehicle conservation (see Equations (4.15) and (4.16)) which is rewritten here using the agent (I) and control step (t) indices.

$$n^{t+1}_{main,I} = n^t_{main,I} + T_s \cdot (q^t_{in,I} + m^t_{r,I} - q^t_{out,I} / (1-\beta_I)) \tag{4.24}$$

$$n^{t+1}_{on,I} = n^t_{on,I} + T_s \cdot (d^t_{on,I} - m^t_{r,I}) \tag{4.25}$$

From the above two equations, it can be seen that $n^{t+1}_{mian,I}$ is determined by $n^t_{main,I}$, $q^t_{in,I}$, $m^t_{r,I}$ and $q^t_{out,I}$, while $n^{t+1}_{on,I}$ can be determined by $n^t_{on,I}$, $m^t_{r,I}$ and $d^t_{on,I}$. According to the fundamental diagram (see Figure 2.2 and Figure 4.3), there exists a relationship between flow ($q^t_{out,I}$) and density ($n^t_{main,I} / l_I$). As the cell length l_I is a fixed value, $q^t_{out,I}$ can be determined by $n^t_{main,I}$. Similar to $q^t_{out,I}$, the on-ramp flow $m^t_{r,I}$ is not an independent variable either. $m^t_{r,I}$ is related to $n^t_{main,I}$, $d^t_{on,I}$ and c^t_I (see Equation (4.14)). Thus, removing $q^t_{out,I}$ and $m^t_{r,I}$, $n^{t+1}_{main,I}$ and $n^{t+1}_{on,I}$ are determined by five variables: $n^t_{main,I}$, $q^t_{in,I}$, $n^t_{on,I}$, $d^t_{on,I}$ and c^t_I. Here, c^t_I is the metering rate generated by ramp metering strategies, which is used to define the control action of a ramp agent. The other four variables are used to define the state.

Among four variables ($n^t_{main,I}$, $q^t_{in,I}$, $n^t_{on,I}$ and $d^t_{on,I}$), $q^t_{in,I}$ and $d^t_{on,I}$ are related to demand flows from the motorway mainline and on-ramp. At each time step, the values of these two variables cannot be changed by control actions. Therefore, the next state at time step $t+1$ (related to $n^{t+1}_{main,I}$, $n^{t+1}_{on,I}$, $q^{t+1}_{in,I}$ and $d^{t+1}_{on,I}$) can be completely determined by the current state (related to $n^t_{main,I}$, $q^t_{in,I}$, $n^t_{on,I}$ and $d^t_{on,I}$) and the control action (related to c^t_I) at time step t. If four state variables related to $n^t_{main,I}$, $q^t_{in,I}$, $n^t_{on,I}$ and $d^t_{on,I}$ are defined as $s^t_{nmain,I}$, $s^t_{qin,I}$, $s^t_{non,I}$ and $s^t_{don,I}$, and the action variable related to c^t_I is defined as a^t_I, the following relationship can be obtained:

$$
\begin{aligned}
&p(s_{nmain,I}^{t+1}, s_{qin,I}^{t+1}, s_{non,I}^{t+1}, s_{don,I}^{t+1} \mid s_{nmain,I}^{0}, s_{qin,I}^{0}, s_{non,I}^{0}, s_{don,I}^{0}, a_I^0, ..., \\
&s_{nmain,I}^{t}, s_{qin,I}^{t}, s_{non,I}^{t}, s_{don,I}^{t}, a_I^t) \\
&= p(s_{nmain,I}^{t+1}, s_{qin,I}^{t+1}, s_{non,I}^{t+1}, s_{don,I}^{t+1} \mid s_{nmain,I}^{t}, s_{qin,I}^{t}, s_{non,I}^{t}, s_{don,I}^{t}, a_I^t)
\end{aligned} \tag{4.26}
$$

The above equation means the state and action at current time step is enough to determine the state at next time step. In other words, the sate and action variables defined here satisfy the Markov property and can be used to formulate an effective RL problem.

Since all states and actions need to be recorded onto the Q (or SQ) table as indices, these states and actions should be converted to integers ranging from 0 to their maximum values. The detailed conversion process will be introduced in Sections 4.3.4 and 4.3.5. All possible values of the state and action can form two sets namely a state set (or state space) S_I and an action set A_I, which can be expressed by:

$$
\begin{cases}
S_{nmain,I} = \{0,1,2,..., N_{nmain,I} - 1\} \\
S_{qin,I} = \{0,1,2,..., N_{qin,I} - 1\} \\
S_{non,I} = \{0,1,2,..., N_{non,I} - 1\} \\
S_{don,I} = \{0,1,2,..., N_{don,I} - 1\} \\
f : S_{nmain,I} \times S_{qin,I} \times S_{non,I} \times S_{don,I} \rightarrow S_I
\end{cases} \tag{4.27}
$$

$$
A_I = \{0,1,2,..., N_{A,I} - 1\} \tag{4.28}
$$

The state set S_I is mapped from four sub-sets $S_{nmain,I}$, $S_{qin,I}$, $S_{non,I}$ and $S_{don,I}$ following a state mapping function f. These four sub-sets contain all possible values of $s_{nmain,I}^t$, $s_{qin,I}^t$, $s_{non,I}^t$ and $s_{don,I}^t$. Similarly, the action set A_I is composed of all possible values of a_I^t which correspond to a number of discrete metering rates within the permitted range. Assuming that the set of discrete metering rates is $C_I = \{c_I(0), c_I(1), c_I(2), ..., c_I(N_{A,I} - 1)\}$, then each action a_I^t of A_I has its own counterpart $c_I(a_I^t)$ in C_I.

4.3.2 Ramp agent structure

Based on the above definition, a ramp agent can be designed to conduct the reinforcement learning process regarding state mapping, reward calculation, action selection, Q value update and scalarisation. All these five functions are incorporated into the agent structure as five sub-modules (as shown in Figure 4.8) belonging to two main modules. The ***RampAgent*** module contains state mapping, action selection and Q values scalarisation. The ***Objective*** module consists of reward calculation and Q value update.

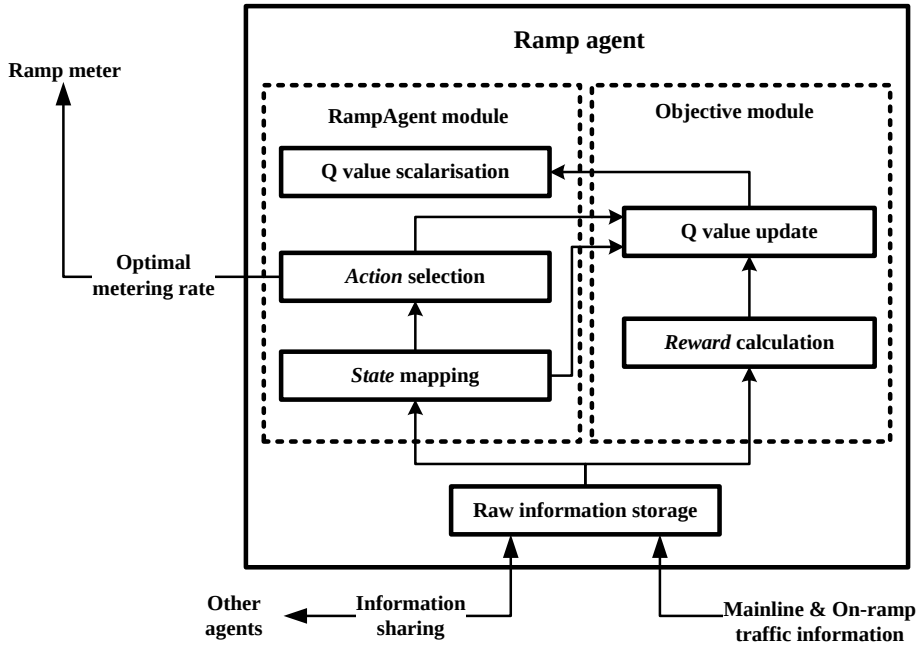


Figure 4.8: Ramp agent structure

For a ramp agent, the state and reward cannot be directly observed from the raw traffic information. Through state mapping and reward calculation, the state of current motorway traffic and the reward for previous executed action can be extracted from the raw information. Based on the extracted information of state, reward and previous recorded Q values, the optimal metering rate can be generated by a suitable action selection strategy. After that, every Q value included (related to each control objective) and the

scalarised Q value aggregated from all Q values can be updated for future use.

The working mechanism of these sub-modules is presented in the following four sections from Section 4.3.3 to 4.3.6.

4.3.3 Reward calculation

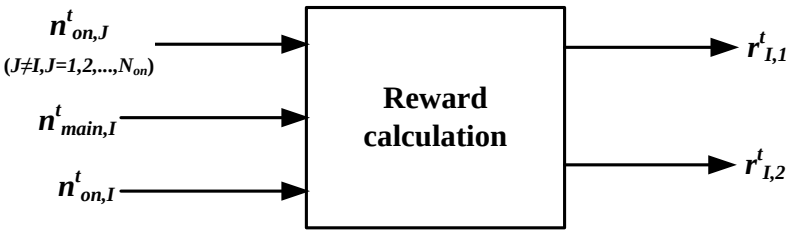


Figure 4.9: Reward calculation

Figure 4.9 illustrates the process of reward calculation which is used to convert raw traffic information regarding agent I (i.e. $n^t_{mian,I}$, $n^t_{on,I}$) and other agents (i.e. $n^t_{on,J}$ $(J \neq I, J = 0,1,2,...,N_{on})$) to the reward at each time step. In Section 4.3.1, two negative rewards have been defined relating to efficiency and equity. For real applications, these two rewards need to be normalised into the same range for the following scalarisation process. In this study, all rewards are normalised into the range [0,1], and a general normalisation process of reward $r^t_{raw,I,j}$ is shown below:

$$r^t_{I,j} = \frac{r^t_{raw,I,j} - r^{\min}_{raw,I,j}}{r^{\max}_{raw,I,j} - r^{\min}_{raw,I,j}} \tag{4.29}$$

where, $r^t_{raw,I,j}$ is the reward calculated directly from the raw traffic information, $r^{\max}_{raw,I,j}$ and $r^{\min}_{raw,I,j}$ are upper and lower bounds for the raw reward value. In this study, the immediate reward received at time step t ($r^t_{I,j}$) refers to the normalised reward.

Efficiency-related reward

For the efficiency-related reward defined by Equation (4.20), the minimum reward value is $r_{raw,I,1}^{\min} = -(n_{main,I}^{\max} + n_{on,I}^{\max})$, and the maximum reward value is $r_{raw,I,1}^{\max}$= 0. Thus, after normalisation, $r_{raw,I,1}^{t}$ can be converted to $r_{I,1}^{t}$ which is expressed by:

$$\begin{aligned} r_{I,1}^{t} &= \frac{r_{raw,I,1}^{t} - r_{raw,I,1}^{\min}}{r_{raw,I,1}^{\max} - r_{raw,I,1}^{\min}} \\ &= \frac{-(n_{main,I}^{t} + n_{on,I}^{t}) - [-(n_{main,I}^{\max} + n_{on,I}^{\max})]}{0 - [-(n_{main,I}^{\max} + n_{on,I}^{\max})]} \\ &= \frac{(n_{main,I}^{\max} + n_{on,I}^{\max}) - (n_{main,I}^{t} + n_{on,I}^{t})}{n_{main,I}^{\max} + n_{on,I}^{\max}} \end{aligned} \tag{4.30}$$

To ensure that $r_{I,1}^{t}$ is strictly in the range [0,1], some conditions should be added into (4.30). Then, the final determined $r_{I,1}^{t}$ is given below.

$$r_{I,1}^{t} = \begin{cases} 0, & \text{if } n_{main,I}^{t} > n_{main,I}^{\max} \text{ or } n_{on,I}^{t} > n_{on,I}^{\max} \\ \dfrac{(n_{main,I}^{\max} + n_{on,I}^{\max}) - (n_{main,I}^{t} + n_{on,I}^{t})}{n_{main,I}^{\max} + n_{on,I}^{\max}}, & \text{otherwise} \end{cases} \tag{4.31}$$

When $n_{main,I}^{t}$ or $n_{on,I}^{t}$ exceeds its maximum permitted value, the smallest reward value 0 will be assigned to $r_{I,1}^{t}$. In this way, it can be guaranteed that $r_{I,1}^{t} \in [0,1]$. In the meanwhile, the on-ramp queue constraints can be considered by setting different values for $n_{on,I}^{\max}$.

Equity-related reward

The equity related reward $r_{raw,I,2}^{t}$ can be normalised according to $r_{raw,I,2}^{\max}$ and $r_{raw,I,2}^{\min}$. When there is no difference between each pair of TWTs, $r_{raw,I,2}^{\max}$ can be obtained that is 0. It has been shown in many studies that the most efficient ramp metering strategy is also the most inequitable strategy (Kotsialos and Papageorgiou 2004a, Meng and Khoo 2010, Zhang and Levinson 2005). Assuming that $SD_{ef}(TWT)$ is the maximum standard

deviation of TWT when efficiency is the only objective considered. Then, $SD_{ef}(TWT)$ should be related to $r_{raw,I,2}^{\min}$. Following the definition of $r_{raw,I,2}^{t}$ in Equation (4.23), it can be obtained that $r_{raw,I}^{\min} = -SD_{ef}(TWT)/T_c$. Given $r_{raw,I,2}^{\max}$ and $r_{raw,I,2}^{\min}$, the normalised $r_{I,2}^{t}$ can be computed with:

$$\begin{aligned} r_{I,2}^{t} &= \frac{r_{raw,I,2}^{t} - r_{raw,I,2}^{\min}}{r_{raw,I,2}^{\max} - r_{raw,I,2}^{\min}} \\ &= \frac{-\sqrt{\sum_{I=1}^{N_{on}} \left(n_{on,I}^{t} - \overline{n}_{on}^{t}\right)^2 / N_{on}} - \left[-SD_{ef}(TWT)/T_c\right]}{0 - \left[-SD_{ef}(TWT)/T_c\right]} \\ &= \frac{SD_{ef}(TWT)/T_c - \sqrt{\sum_{I=1}^{N_{on}} \left(n_{on,I}^{t} - \overline{n}_{on}^{t}\right)^2 / N_{on}}}{SD_{ef}(TWT)/T_c} \end{aligned} \tag{4.32}$$

Similar to $r_{I,1}^{t}$ defined in Equation (4.31), some conditions should be added into (4.32) to guarantee that $r_{I,2}^{t} \in [0,1]$. Then, $r_{I,2}^{t}$ is rewritten as:

$$r_{I,2}^{t} = \begin{cases} 0, & \text{if } SD_{ef}(TWT)/T_c \le \sqrt{\sum_{I=1}^{N_{on}} \left(n_{on,I}^{t} - \overline{n}_{on}^{t}\right)^2 / N_{on}} \\ \dfrac{SD_{ef}(TWT)/T_c - \sqrt{\sum_{I=1}^{N_{on}} \left(n_{on,I}^{t} - \overline{n}_{on}^{t}\right)^2 / N_{on}}}{SD_{ef}(TWT)/T_c}, & \text{otherwise} \end{cases} \tag{4.33}$$

4.3.4 State mapping

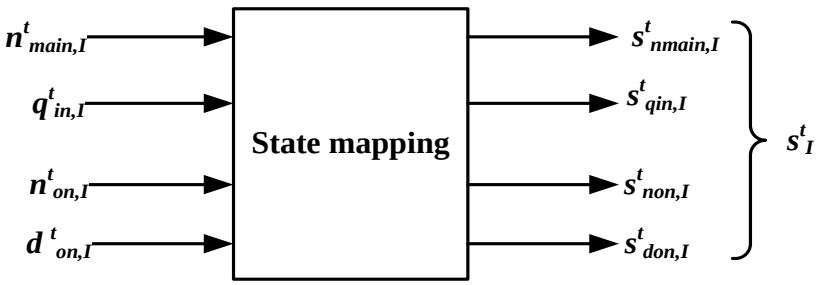


Figure 4.10: State mapping

As shown in Figure 4.10, state mapping is a translation process, through which the raw traffic information collected from the controlled motorway ($n_{main,I}^{t}$, $q_{in,I}^{t}$, $n_{on,I}^{t}$, $d_{on,I}^{t}$) can be translated to state variables ($s_{main,I}^{t}$, $s_{in,I}^{t}$, $s_{on,I}^{t}$,

$s_{on,I}^t$) required by the ramp agent. The final determined state s_I^t belonging to the state set S_I is calculated from these four state variables. Functions regarding the state mapping are given in Equations (4.34) to (4.39):

$$s_{nmain,I}^t = \begin{cases} 0, & \text{if } n_{main,I}^t \le n_{main,I}^{low} \\ \left\lceil (n_{main,I}^t - n_{main,I}^{lower}) / \Delta n_{main,I} \right\rceil, & \text{if } n_{main,I}^{low} < n_{main,I}^t \le n_{main,I}^{up} \\ \left\lceil (n_{main,I}^{up} - n_{main,I}^{low}) / \Delta n_{main,I} \right\rceil + 1, & \text{otherwise} \end{cases} \tag{4.34}$$

$$s_{qin,I}^t = \begin{cases} 0, & \text{if } q_{in,I}^t \le q_{in,I}^{low} \\ \left\lceil (q_{in,I}^t - q_{in,I}^{low}) / \Delta q_{in,I} \right\rceil & \text{if } q_{in,I}^{low} < q_{in,I}^t \le q_{in,I}^{up} \\ \left\lceil (q_{in,I}^{up} - q_{in,I}^{low}) / \Delta q_{in,I} \right\rceil + 1, & \text{otherwise} \end{cases} \tag{4.35}$$

$$s_{non,I}^t = \begin{cases} 0, & \text{if } n_{on,I}^t \le n_{on,I}^{low} \\ \left\lceil (n_{on,I}^t - n_{on,I}^{low}) / \Delta n_{on,I} \right\rceil & \text{if } n_{on,I}^{low} < n_{on,I}^t \le n_{on,I}^{up} \\ \left\lceil (n_{on,I}^{up} - n_{on,I}^{low}) / \Delta n_{on,I} \right\rceil + 1, & \text{otherwise} \end{cases} \tag{4.36}$$

$$s_{don,I}^t = \begin{cases} 0, & \text{if } d_{on,I}^t \le d_{on,I}^{low} \\ \left\lceil (d_{on,I}^t - d_{on,I}^{low}) / \Delta d_{on,I} \right\rceil & \text{if } d_{on,I}^{low} < d_{on,I}^t \le d_{on,I}^{up} \\ \left\lceil (d_{on,I}^{up} - d_{on,I}^{low}) / \Delta d_{on,I} \right\rceil + 1, & \text{otherwise} \end{cases} \tag{4.37}$$

$$\begin{cases} N_{nmain,I} = \left| S_{nmain,I} \right| = \left\lceil (n_{main,I}^{up} - n_{main,I}^{low}) / \Delta n_{main,I} + 2 \right\rceil \\ N_{qin,I} = \left| S_{qin,I} \right| = \left\lceil (q_{in,I}^{up} - q_{in,I}^{low}) / \Delta q_{in,I} + 2 \right\rceil \\ N_{non,I} = \left| S_{non,I} \right| = \left\lceil (n_{on,I}^{up} - n_{on,I}^{low}) / \Delta n_{on,I} + 2 \right\rceil \\ N_{don,I} = \left| S_{don,I} \right| = \left\lceil (d_{on,I}^{up} - d_{on,I}^{low}) / \Delta d_{on,I} + 2 \right\rceil \end{cases} \tag{4.38}$$

$$s_I^t = N_{qin,I} \cdot N_{non,I} \cdot N_{don,I} \cdot s_{nmain,I}^t + N_{non,I} \cdot N_{don,I} \cdot s_{qin,I}^t + N_{don,I} \cdot s_{non,I}^t + s_{don,I}^t \tag{4.39}$$

In the above equations, $\lceil\ \rceil$ is the ceiling function that returns the smallest integer not less than the considered value. Through Equations (4.34) to (4.37), all state sets can be discretised into a number of states with integer

values, and all raw traffic information can be mapped to its corresponding states.

Take Equation (4.34) for example, within the predefined boundaries $n_{main,I}^{low}$ (usually 0) and $n_{main,I}^{up}$ (usually $n_{main,I}^{\max}$), the maximum number of vehicles on the mainline can be uniformly divided into a group of intervals according to $\Delta n_{main,I}$. Each interval corresponds to a state with a value from 1 to $\left\lceil (n_{main,I}^{up} - n_{main,I}^{low}) / \Delta n_{main,I} \right\rceil$. When $n_{main,I}^{t}$ exceeds two boundary values, two additional states 0 and $\left\lceil (n_{main,I}^{up} - n_{main,I}^{low}) / \Delta n_{main,I} + 1 \right\rceil$ are added into the state set. Hence, $S_{nmain,I}$ has $N_{nmain,I} = \left\lceil (n_{main,I}^{up} - n_{main,I}^{low}) / \Delta n_{main,I} + 2 \right\rceil$ states, and $S_{nmain,I} = \{0,1,2,...,N_{nmain} - 1\}$. Through Equation (4.34), at time step t, $n_{main,I}^{t}$ can be mapped to $s_{nmain,I}^{t}$ that corresponds to a value of $\{0,1,2,...,N_{nmain} - 1\}$.

In the same way, other raw traffic information related to $q_{in,I}^{t}$, $n_{on,I}^{t}$ and $d_{on,I}^{t}$ can be mapped to $s_{qin,I}^{t}$, $s_{non,I}^{t}$ and $s_{don,I}^{t}$ through Equations (4.35)~(4.37). State numbers $N_{qin,I}$, $N_{non,I}$ and $N_{don,I}$ can be obtained by (4.38). Thus, $S_I = \{0,1,2,...,N_{nmain} \cdot N_{qin,I} \cdot N_{non,I} \cdot N_{don,I} - 1\}$. By combining all these four state variables, s_I^t can be obtained by Equation (4.39), which is a value of $\{0,1,2,...,N_{nmain} \cdot N_{qin,I} \cdot N_{non,I} \cdot N_{don,I} - 1\}$.

4.3.5 Action selection

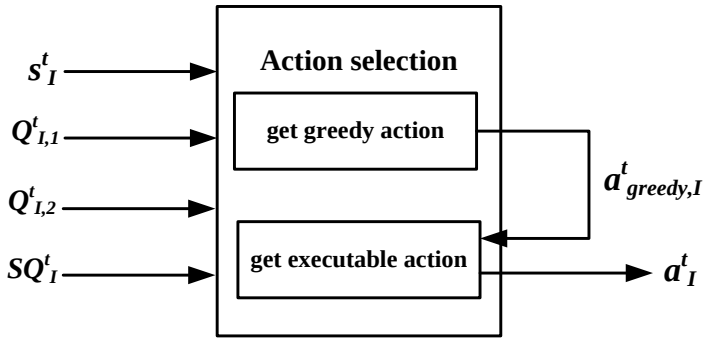


Figure 4.11: Action selection

As shown in Figure 4.11, action selection is responsible for selecting suitable actions at each state according to the recorded Q ($Q_{I,1}^t$ related to $r_{I,1}^t$, and $Q_{I,2}^t$ related to $r_{I,2}^t$) and SQ (scalarised Q) values. According to the Q-learning mechanism introduced in Section 3.4.1, two kinds of actions, the greedy action $a_{greedy,I}^t$ and the executable action a_I^t should be selected by the agent for Q values update and real execution, respectively.

Recall that in Section 4.3.1, the action set A_I has been defined for a ramp metering problem, which corresponds to a group of discrete metering rates. The discretisation can be conducted using a method proposed by (Kotsialos et al. 2006) where the metering rate is uniformly divided into a number of integer values. Then, the relationship between the action variable and its corresponding discrete metering rate can be derived from:

$$c_I(a_I^t) = c_I^{\min} + \frac{a_I^t}{N_{A,I} - 1} \cdot (c_I^{\max} - c_I^{\min}) \tag{4.40}$$

where, $c_I^{\max}$ and $c_I^{\min}$ are the maximum and minimum permitted values of metering rate. In this way, each discrete metering rate $c_I(a_I^t)$ is related to an action a_I^t. For real applications, $N_{A,I}$ can be regulated to ensure that all discrete metering rates are integer values.

The greedy action $a_{greedy,I}^t$ corresponds to the optimal metering rate which is determined according to current Q or SQ values. If traffic efficiency is the only objective considered, the greedy action can be determined by related Q values as shown below:

$$a_{greedy,I}^t = \arg\max_{a_I^t}(Q_{I,1}^{t-1}(s_I^t, a_I^t)) \tag{4.41}$$

If both traffic efficiency and user equity should be considered, the greedy action is determined by SQ values.

$$a^t_{greedy,I} = \arg\max_{a^t_I}(SQ^{t-1}_I(s^t_I, a^t_I)) \tag{4.42}$$

For real execution, an exploration probability should be considered to explore non-greedy actions. If ε-greedy strategy is used (as introduced in Section 3.4.2), the executable action a^t_I can be selected according to the following probability:

$$p(a^t_I \mid s^t_I) = \begin{cases} \varepsilon, & \text{if } a^t_I \neq a^t_{greedy,I} \\ 1-\varepsilon, & \text{otherwise} \end{cases} \tag{4.43}$$

A relationship between the control and simulation is introduced in Section 4.2.4 (Figure 4.7), which defines that each control interval may contain several simulation steps. Under such regulation, the control action at one control step may be used to determine the metering rates for more than one simulation step. To correlate action a^t_I of the ramp agent and metering rate c^k_I of ACTM, the following relationship should be satisfied:

$$c^k_I = \frac{c_I(a^t_I)}{N_{cs}} \tag{4.44}$$

where, t is the control step and k is the simulation step. At each control step, a value $a^t_I \in A_I$ is selected as the ramp metering rate and evenly assigned to each simulation step belonging to this control step.

4.3.6 Q value update and scalarisation

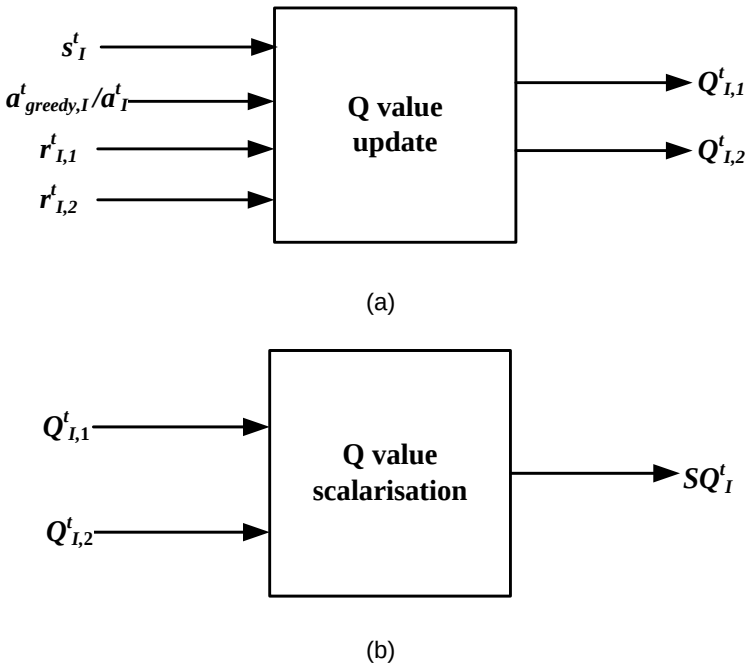


Figure 4.12: Q update and scalarisation

The ramp agent can obtain the immediate feedback from a controlled motorway by converting the raw traffic information to rewards. The long-term impacts of its actions are recorded by Q values (cumulative rewards) which are the discounted aggregation of rewards observed at different time steps.

As shown in Figure 4.12 (a), two Q tables regarding efficiency ($Q_{I,1}$) and equity ($Q_{I,2}$) are maintained and updated for each recorded state and action pair. For a Q-learning problem, the updating rule given by Equation (3.11) can be used to update the Q table at each time step. For ramp agent I designed in this chapter, this updating rule can be rewritten as the following two equations with the agent index I:

$$Q_{I,1}{}^{t}(s_I{}^{t-1}, a_I{}^{t-1}) = Q_{I,1}{}^{t-1}(s_I{}^{t-1}, a_I{}^{t-1}) + \alpha_I \cdot [r_{I,1}{}^{t} + \gamma_I \cdot Q^{t-1}_{I,1}(s_I{}^{t}, a^t_{greedy,I}) - Q_{I,1}{}^{t-1}(s_I{}^{t-1}, a_I{}^{t-1})] \quad (4.45)$$

$$Q_{I,2}{}^{t}(s_I{}^{t-1}, a_I{}^{t-1}) = Q_{I,2}{}^{t-1}(s_I{}^{t-1}, a_I{}^{t-1}) + \alpha_I \cdot [r_{I,2}{}^{t} + \gamma_I \cdot Q^{t-1}_{I,2}(s_I{}^{t}, a^t_{greedy,I}) - Q_{I,2}{}^{t-1}(s_I{}^{t-1}, a_I{}^{t-1})] \quad (4.46)$$

Equation (4.45) is used to update $Q_{I,1}$, while Equation (4.46) corresponds to the updating of $Q_{I,2}$.

For the multi-objective problem, different Q values regarding different control objectives should be aggregated together through linear scalarisation as shown in Figure 4.12 (b). In this study, only two control objectives (regarding $Q_{I,1}$ and $Q_{I,2}$) are included in the ramp metering problem. According to the linear scalarisation method introduced in Section 3.5.3 (Equation (3.13)), the scalarised Q value for the two-objective case can be obtained by:

$$SQ_I^t(s_I^{t-1}, a_I^{t-1}) = \delta_1 \cdot Q_{I,1}^t(s_I^{t-1}, a_I^{t-1}) + \delta_2 \cdot Q_{I,2}^t(s_I^{t-1}, a_I^{t-1}) \quad (4.47)$$

where, δ_1 is the weight value for $Q_{I,1}$, δ_2 is the weight value for $Q_{I,2}$, and $\delta_1 + \delta_2 = 1$.

4.3.7 Summary

Section 4.3 presented the systematic design of a ramp agent including the definition of three basic elements (i.e. state, action and reward) and the working mechanism of five sub-modules (i.e. state mapping, reward calculation, action selection, Q value update and scalarisation).

Among three basic elements, the reward was defined according to the control objectives considered. In this section, two kinds of rewards were derived from the definitions of two objectives regarding traffic efficiency and user equity. The state and action were defined to satisfy the Markov property, which guaranteed that the definitions in this section can formulate an effective RL problem.

Based on the element definition, a structure of ramp agent containing five sub-modules was proposed. State mapping and reward calculation can

convert raw traffic information to states and rewards required by the ramp agent. Action selection can help the agent select suitable metering rates to control the motorway. Q value update and scalarisation are responsible for updating the agent's memory about the optimal metering rates. These five sub-modules contain all the information required by a learning process, which will be used to develop two learning algorithms in the next section.

4.4 Algorithms of Ramp Agent

Two modes of RAS have been introduced in Section 4.1.2 including the single-objective mode and the multi-objective mode. This section firstly presents an algorithm for the single-objective mode. Then this algorithm is extended to a more general form that can deal with multiple objectives. Although the main aim of algorithms developed in this section is to deal with two main control objectives regarding improving traffic efficiency (i.e. reducing TTS) and maintaining user equity (i.e. balancing TWT), other objectives can be involved by using the same framework of these two algorithms.

4.4.1 Single-objective algorithm

When reducing TTS is the only concern and no information from other agents is required, a single-objective algorithm can be used to learn the optimal solution for ramp metering control. In this case, the ramp agent only captures the information within its vicinity and tries to obtain the minimum total time spent of its controlled motorway segment. The flow chart of this single-objective algorithm is shown in Figure 4.13.

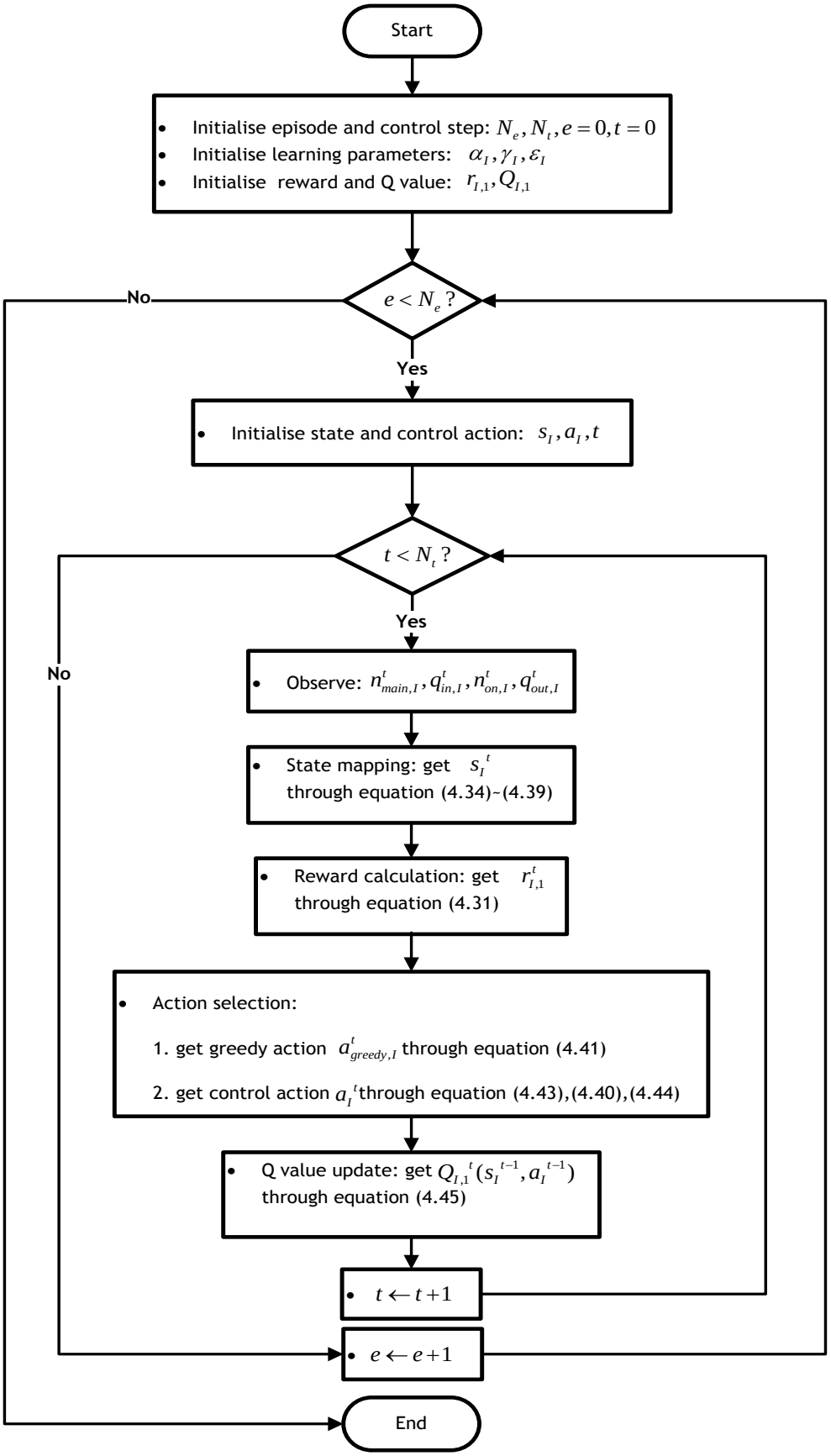


Figure 4.13: Flow chart for the single-objective algorithm

The basic single-objective algorithm contains two loops for episode e and control step t respectively. One episode (or iteration) represents one whole control period that usually begins with the occurrence of congestion and ends when the traffic returns to its normal situation (e.g. the algorithm can be triggered during the peak hours). Each episode may consist of a sequence of control steps, the number of which depends on the duration of the whole control period and each control interval. The number of episodes is determined according to experience, which is usually set as a big enough number to guarantee that the ramp agent can learn the optimal strategy after finishing these episodes.

The algorithm starts with the initialisation of relevant parameters for the learning strategy and control objective. Then, for each episode, the initial state and action should be set for the loop of control step. At each control step, the ramp agent obtains the current state of its controlled motorway segment through state mapping and receives the reward for its executed action at last step by reward calculation. After that, the ramp agent selects the greedy action for the current state and takes one executable action to perform by using an action selection strategy (here it is ε-greedy strategy). Finally, the Q value for observed state-action pair recorded in the Q table is updated.

4.4.2 Multi-objective algorithm

The single-objective algorithm introduced above can only deal with local objective problems (with local information within the vicinity of one ramp agent). For more general scenarios which consider both local (can be finished by one ramp agent) and group (need to be achieved by a group of agents) objectives, a multi-objective algorithm is developed in Figure 4.14.

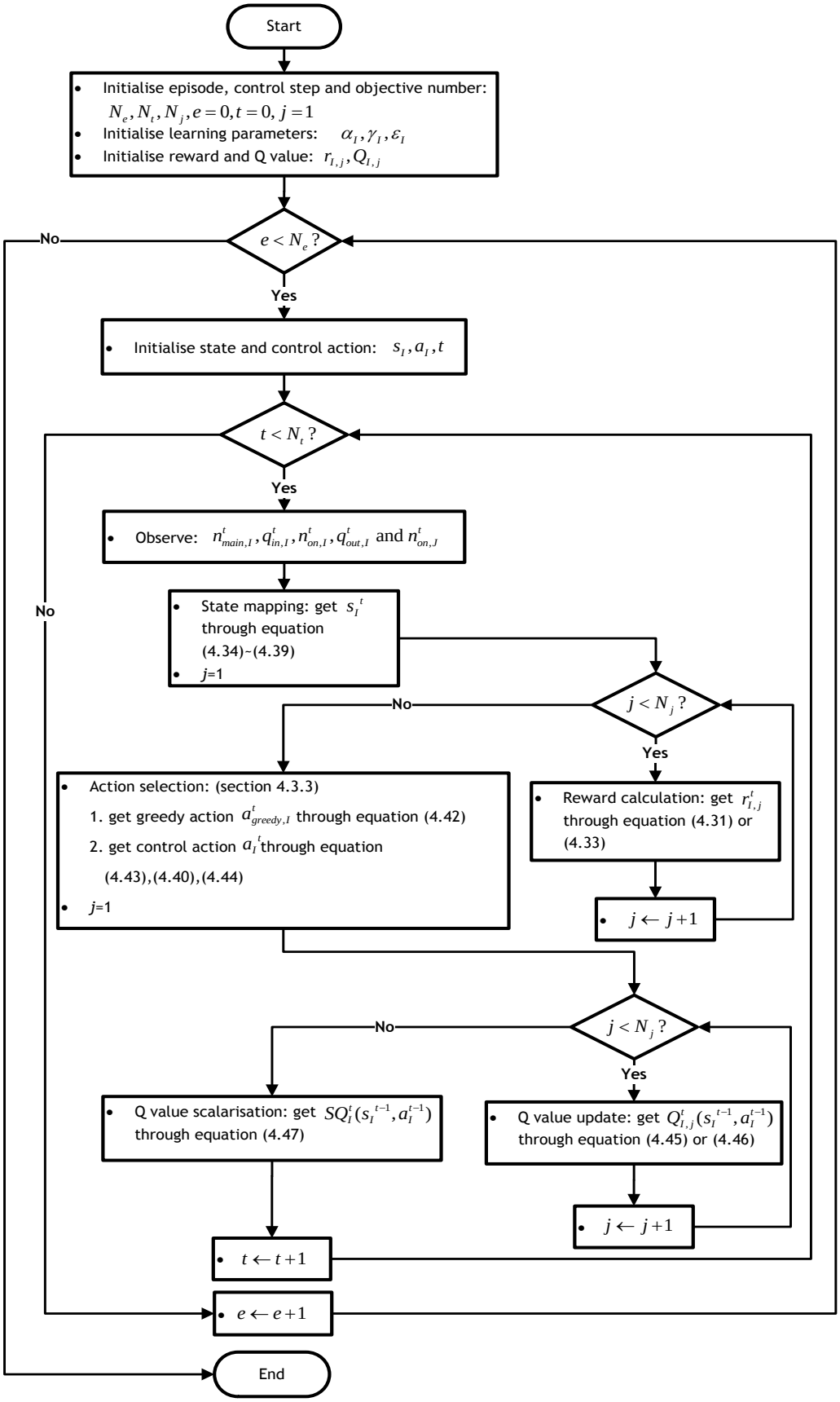


Figure 4.14: Flow chart for the multi-objective algorithm

Similar to the algorithm introduced in the previous section, the multi-objective algorithm also contains two main loops for the episode and control step. To deal with multiple objectives, two additional loops for different objectives (j) are embedded in the algorithm.

The multi-objective algorithm also begins with the initialisation of related parameters. Similar to the single-objective algorithm, the ramp agent starts with observing the state and receiving reward at each control step. The only difference is the reward related to each objective should be recorded separately. Then the ramp agent selects actions for the Q update and execution according to the SQ value, not the Q value in the single-objective case. After that, Q values are updated and scalarised to obtain the new SQ value for the recorded state-action pair. The Q scalarisation function should be used to calculate the SQ value as a weighted sum of different Q values for different objectives.

4.4.3 Ending rules of algorithms

In the above descriptions, algorithms end after a fixed number of episodes. If the target value of each control objective is known in advance, an alternative ending rule may be used to end the algorithm. According to these target values, two boundaries $TTS_{boundary,I}$ and $SD(TWT_I)_{boundary}$ can be set for TTS and SD(TWT), respectively. The algorithm does not need to finish all episodes and can end as long as the predefined boundaries are achieved. One example of this ending rule is shown in Figure 4.15. The algorithm will end when $TTS_I \leq TTS_{boundary,I}$. In the same way, $SD(TWT_I) \leq SD(TWT_I)_{boundary}$ can be used as another condition to determine whether the algorithm should end.

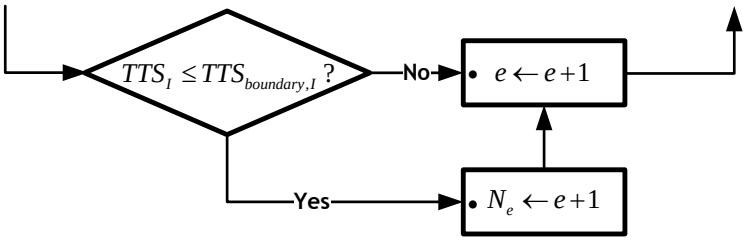


Figure 4.15: An example of ending rule

Usually, in the first run of the algorithm, a fixed number of episodes are required to obtain the optimal objective value. Then, this value can be set as a boundary of the algorithm for the following runs with ending rules introduced in this section.

4.4.4 Summary

In this section, two learning algorithms were developed to deal with single-objective and multi-objective problems. In the single-objective mode, only traffic efficiency was considered and the ramp agent was only concerned with what happened on its own controlled motorway segment.

On the other hand, two objectives including both efficiency and equity were added into the multi-objective mode, which required the information from other agents. Although only two control objectives of minimising TTS and keeping equity were considered in the current stage, more objectives may be added under the same algorithm framework (shown in Figure 4.14).

4.5 Discussion

In this chapter, a systematic design of ramp agent that can deal with ramp metering control problems was provided. A traffic flow model, ACTM, was used to simulate the motorway traffic and evaluate the ramp agent system (RAS).

Some issues related to the traffic flow model and the main contributions of this chapter are highlighted here.

(1) The discontinuous version of ACTM was selected for evaluation because of its capability of mimicking traffic dynamics on motorways especially the capacity drop phenomenon. This model is independent of RAS. Under the general architecture of agent-environment interaction, RAS can be evaluated by any simulation models or even the real motorway traffic. That is why RAS is considered as a model-free method.

(2) In this chapter, a general definition of three basic elements, i.e. state, action and reward was proposed for ramp metering problems. The definition of two rewards was derived from a general objective function in the ramp metering domain that is related to TTS and TWT. The definition of state and action was based on the vehicle conservation and satisfied the Markov property. Such a general definition satisfying the Markov property was omitted by previous related studies introduced in Section 2.3.4. Each study had its own way of defining the three elements, especially the reward and state, which cannot guarantee their effectiveness. The general definition proposed in this chapter provided a clearer way to define these elements based on the Markov property, and thus, they can guarantee an effective RL process.

(3) A group of sub-modules for the RL, or specifically Q-learning process was developed on the basis of a general definition of three elements. These sub-modules contained all the information required to accomplish a Q-learning process and were used to formulate two algorithms regarding single- and multi-objective problems. Although two specific objectives are considered at the current stage, these two algorithms can be extended to include other objectives based on their general frameworks shown in Sections 4.4.1 and 4.4.2.

After the whole design process of a ramp agent system presented in this chapter, Chapter 5 will introduce the relevant software implementation issues.

CHAPTER 5 IMPLEMENTATION

Chapter 4 described how to design a ramp agent system and details of a macroscopic traffic flow model “ACTM” used for evaluation. The next problem is how to implement these models and use them as a platform to do the evaluation. In this chapter, the software implementation of ACTM and ramp agent system with reusable classes are developed, which provides a flexible way to evaluate the RL-based ramp metering system under different traffic conditions. The full source code (including header files and source files) is shown in Appendix A, this chapter will focus on the general implementation issue and some main functions related to it.

5.1 ACTM Implementation

The implementation issue of this section is focused on using C++ to implement ACTM introduced in Section 4.2. ACTM is programmed by Visual C++ 6.0 in this work.

5.1.1 Class diagram

The cell of ACTM is classified into four categories corresponding to normal cell, on-ramp cell, off-ramp cell and on-off cell, as defined in Section 4.2.2. Examples of four kinds of layout for different cells can be seen form Figure 5.1. For real applications, different combinations of these four cells can be chosen by software users to simulate different motorway networks with different layouts.

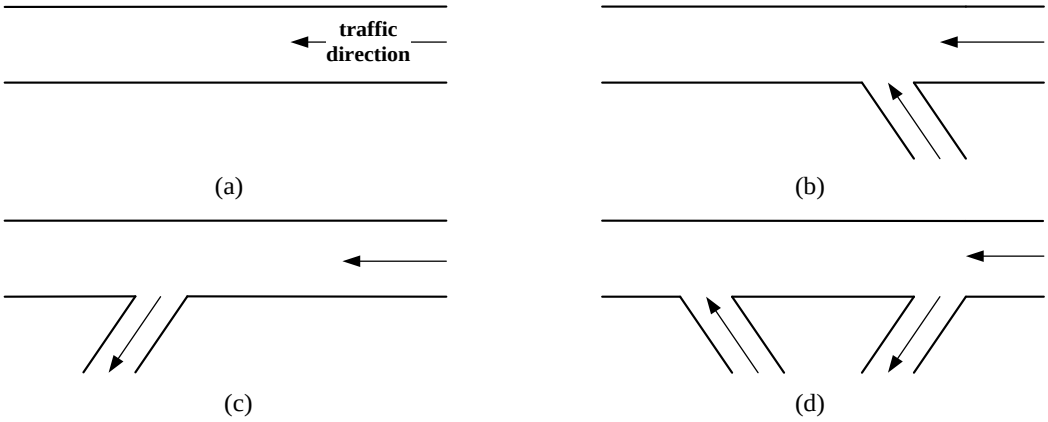


Figure 5.1: Examples of different cells: (a) normal cell, (b) on-ramp cell, (c) off-ramp cell, (d) on-off cell

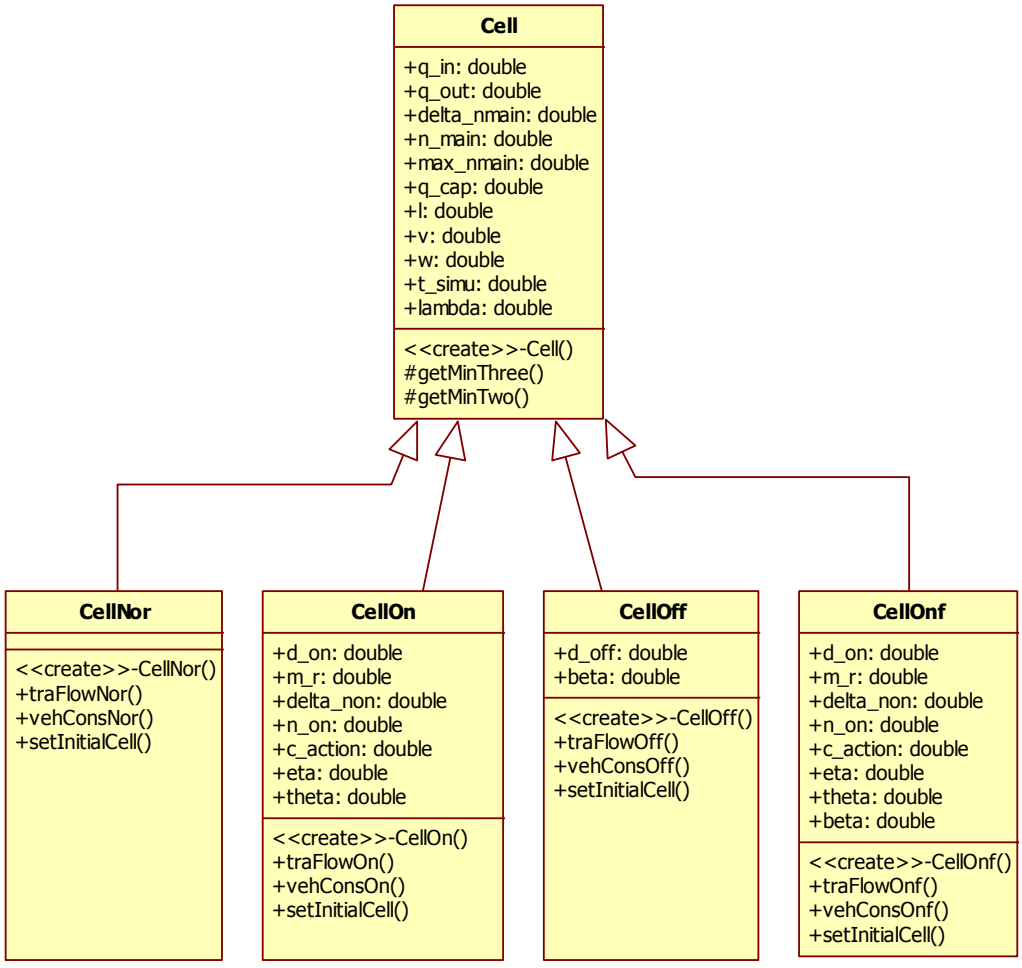


Figure 5.2: Class diagram for ACTM

To represent four kinds of cells in the program, four classes: ***CellNor*** (normal cell), ***CellOn*** (on-ramp cell), ***CellOff*** (off-ramp cell) and ***CellOnf*** (on-off cell) are developed (see Figure 5.2). These four classes inherit from one parent class: ***Cell***. The class ***Cell*** contains basic variables and functions related to all kinds of cells. For each kind of cell, two specific functions are used to calculate traffic flow dynamics and vehicle conservation.

5.1.2 Functions for flow dynamics

Flow dynamics mentioned here refer to the calculation of mainline flow and ramp flow defined by Equations (4.14) and (4.17) (in Sections 4.2.2 and 4.2.3). The functions related to flow dynamics are used to solve (4.14) and (4.17) at each simulation step.

The class ***CellNor*** does not contain any additional variables for on- and off-ramps, so the traffic flow dynamics are only related to mainline traffic flows. The function responsible for calculating flow dynamics on the mainline is named ***traFlowNor()***, which is shown in Appendix A.1, code 5. A Boolean variable ***capa_drop*** is used in this function to determine whether the capacity drop phenomenon is considered.

For the on-ramp cell (class ***CellOn***), one on-ramp is linked with the mainline. Thus, flow dynamics are composed of mainline flow and on-ramp flow dynamics. The implementation of ***traFlowOn()*** is shown in Appendix A.1, code 9. An additional Boolean variable ***control*** is used by ramp metering strategies, which determines whether or not the cell is under control.

The class ***CellOff*** contains functions and variables related to the mainline and off-ramp traffic. On-ramp flow is not considered in this class. The function ***traFlowOff()*** is shown in Appendix A.1, code 13.

Both on- and off-ramps are considered in the class ***CellOnf***, which is a combination of on-ramp cell and off-ramp cell. Here, the function

traFlowOnf() is responsible for calculating traffic flows in all three areas: mainline, on-ramp and off-ramp. The implementation of this function is given in Appendix A.1, code 17.

5.1.3 Functions for vehicle conservation

As shown in Equations (4.15) and (4.16) (in Section 4.2.2), vehicle conservation contains the conservation on the mainline and on-ramp. Functions related to conservation are summarised below.

The function related to the class ***CellNor*** is named as ***vehConsNor()***, which only considers the mainline traffic conservation of a normal cell and is given in Appendix A.1, code 6.

For the class ***CellOn***, both mainline and on-ramp conservation should be considered. The function ***vehConsOn()*** used to calculate the conservation of an on-ramp cell is shown in Appendix A.1, code 10.

The conservation function ***vehConOff()*** of the class ***CellOff*** is similar to ***vehConsNor()***, except that the split flow (related to split ratio ***beta***) should be considered in the mainline conservation of an off-ramp cell. This function is shown in Appendix A.1, code 14.

The conservation function ***vehConsOnf()*** of the class ***CellOnf*** is a combination of related functions from the classes ***CellOn*** and ***CellOff***, which is implemented in Appendix A.1, code 18.

For real applications, different cells can be instantiated according to the layout of motorway network. Two functions related to traffic flow dynamics and vehicle conservation can be called to simulate traffic operations represented by Equations (4.14) and (4.17).

In addition to the main functions introduced above, some other functions are also embedded in ACTM. These functions are responsible for basic

calculations and initialisation required by the main functions. The annotation of these functions is listed in Table 5.1.

Table 5.1: Function annotations of ACTM

Function	Class	Annotation
Cell()	*Cell*	constructor used for initialisation
getMinThree()	*Cell*	get the minimum value of three arguments
getMinTwo()	*Cell*	get the minimum value of two arguments
setInitialCell()	*CellNor* *CellOn* *CellOff* *CellNof*	set initial parameters for cells

5.2 Ramp Agent Implementation

Section 4.3 has introduced main modules of a ramp agent including state mapping, reward calculation, action selection, Q value update and scalarisation. This section will discuss how to convert them into reusable classes and functions.

5.2.1 Class diagram

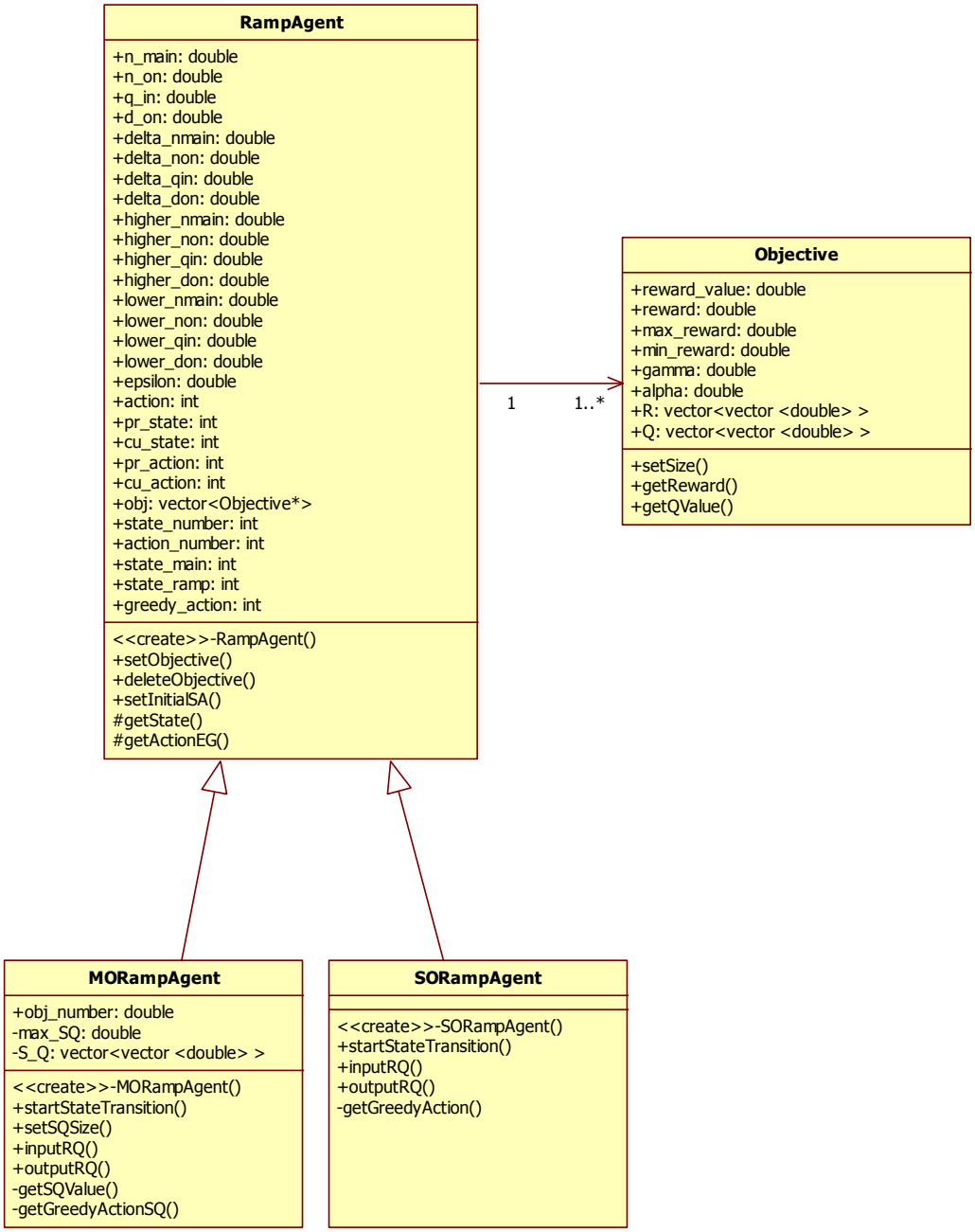


Figure 5.3: Class diagram for the ramp agent

Figure 5.3 shows the class diagram for a ramp agent. There are four classes in the agent architecture including ***RampAgent***, ***SORampAgent***, ***MORampAgent*** and ***Objective***. The class ***RampAgent*** contains the main functions related to the application of RL such as state mapping, action selection and Q value scalarisation. Both the classes ***SORampAgent*** and ***MORampAgent*** are inherited from the class ***RampAgent*** and designed according to different modes. To be more specific, ***SORampAgent*** is designed for single-objective mode only, while ***MORampAgent*** can deal with multi-objective problems with more than one objective. The class ***Objective*** maintains a table recording the Q value for each state-action pair. Functions for reward calculation and updating Q values are also contained in this class. These functions help the ramp agent accomplish its learning process according to different control objectives, which are the software implementations of two control algorithms introduced in Section 4.4, or specifically the Equations (4.29) to (4.47). The main functions related to ramp agent and its control objectives will be described in detail in the next two sections.

5.2.2 Functions for ramp agent

The state mapping of a ramp agent is realised by the function ***getState()***. This function is responsible for getting the state according to real-time data observed from road traffic (refer to Equations (4.34) to (4.39) in Section 4.3.4). The source code of this function is shown in Appendix A.2, code 5.

The action selection module is responsible for selecting two kinds of actions: the greedy action and executable action, which can be realised by three functions introduced as follows.

For ***SORampAgent***, the greedy action at each state is selected according to Q values and can be implemented by ***getGreedyAction()*** (refer to Equation

(4.41) in Section 4.3.5). The implementation of this function is shown in Appendix A.2, code 9.

The greedy action of ***MORampAgent*** is selected according to SQ values not Q values, which can be obtained by ***getGreedyActionSQ()*** (refer to Equation (4.42) in Section 4.3.5). This function is shown in Appendix A.2, code 17.

Based on the greedy action, the real executable action (using ε-greedy strategy) can be obtained by the function ***getActionEG()*** (refer to Equation (4.43) in Section 4.3.5). This function is contained in the class ***RampAgent*** which can be used by both the classes ***SORampAgent*** and ***MORampAgent***. The source code of this function can be found in Appendix A.2, code 6.

Another module of the ramp agent is Q scalarisation, which can be realised by ***getSQValue()*** (refer to Equation (4.47) in Section 4.3.5). This function is responsible for scalarising Q values to get the SQ value. The source of this function is shown in Appendix A.2, code 18.

5.2.3 Functions for objective

The class ***Objective*** contains two main functions ***getReward()*** and ***getQValue()***, which can be used to realise the reward calculation and Q values update, respectively.

The function ***getReward()*** is used to get the immediate reward according to real-time data observed from road traffic (refer to Equations (4.29) to (4.33) in Section 4.3.3). The function ***getQValue()*** is responsible for getting the Q value for each state-action pair (refer to Equations (4.45) and (4.46) in Section 4.3.5). These two functions are implemented in Appendix A.3, code 3 and code 5, respectively.

Besides the main functions related to the learning process, some other functions are included in RAS for the basic operations, such as initialising the system, maintaining Q tables and objectives. The annotation of these functions is shown in Table 5.2.

Table 5.2: Function annotations for RAS

Function	Class	Annotation
RampAgent()	*RampAgent*	constructor used for initialisation
setInitialSA()	*RampAgent*	set the initial state and action
setObjective()	*RampAgent*	set objectives according to the number of objectives required
deleteObjective()	*RampAgent*	delete objectives
SORampAgent()	*SORampAgent*	constructor used for initialisation
MORampAgent()	*MORampAgent*	constructor used for initialisation
setSQSize()	*MORampAgent*	set the scalarised Q table size
setSize()	*Objective*	set the R and Q table size
startStateTransition()	*SORampAgent* *MORampAgent*	do the state transition within the state space which combines all functions to accomplish the learning process
inputRQ()	*SORampAgent* *MORampAgent*	read Q values from a text file
outputRQ()	*SORampAgent* *MORampAgent*	write Q table to a text file
setSize()	*Objective*	set the Q table size

5.2.4 Sequence diagrams

The class diagram has shown the main functions and variables required by the ***RampAgent*** class and the ***Objective*** class. This section will introduce how these two classes work together to conduct the learning process. Two sequence diagrams related to ***SORampAgent*** and ***MORampAgent*** are shown in Figures 5.4 and 5.5, respectively.

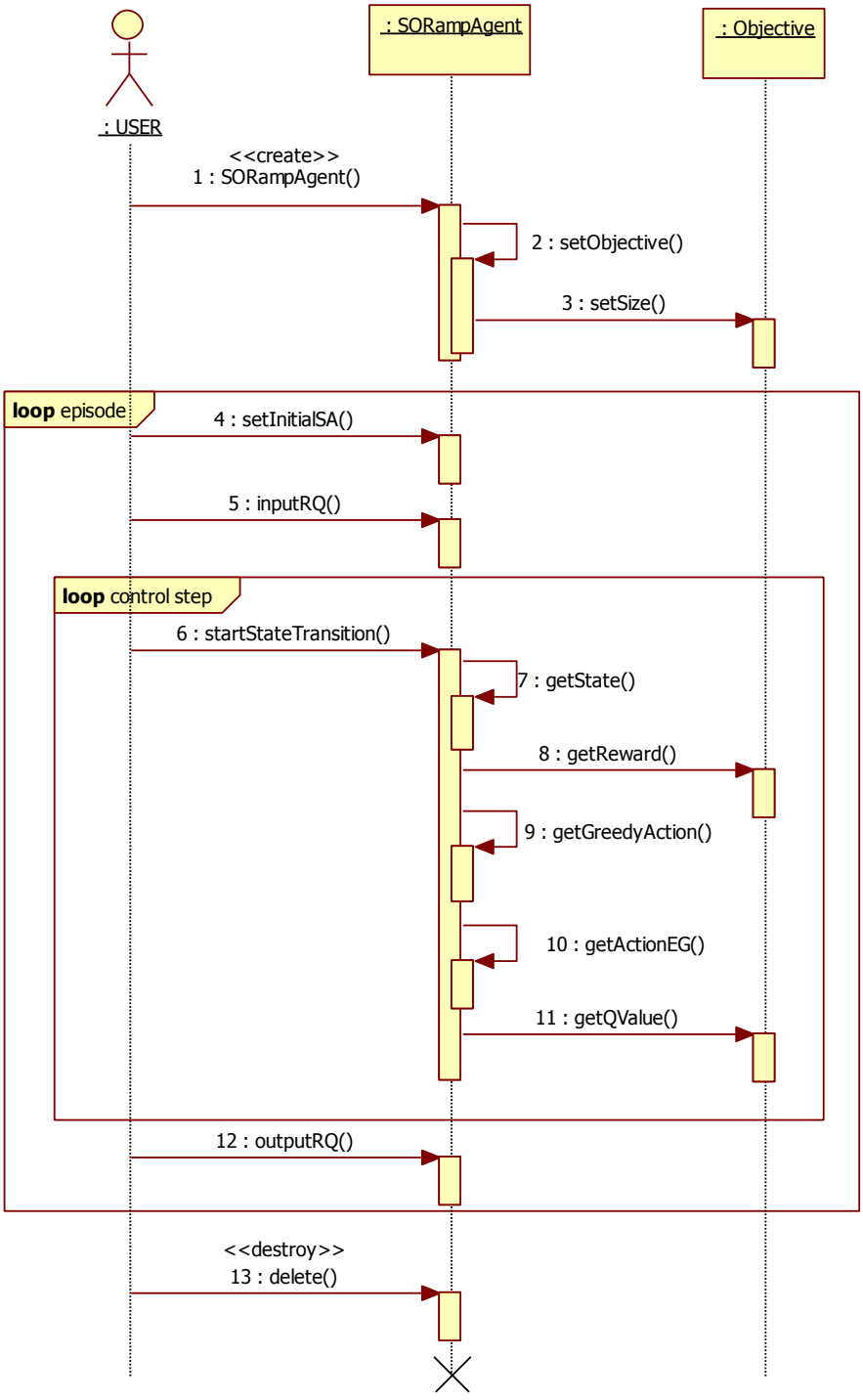


Figure 5.4: Sequence diagram for the single-objective mode

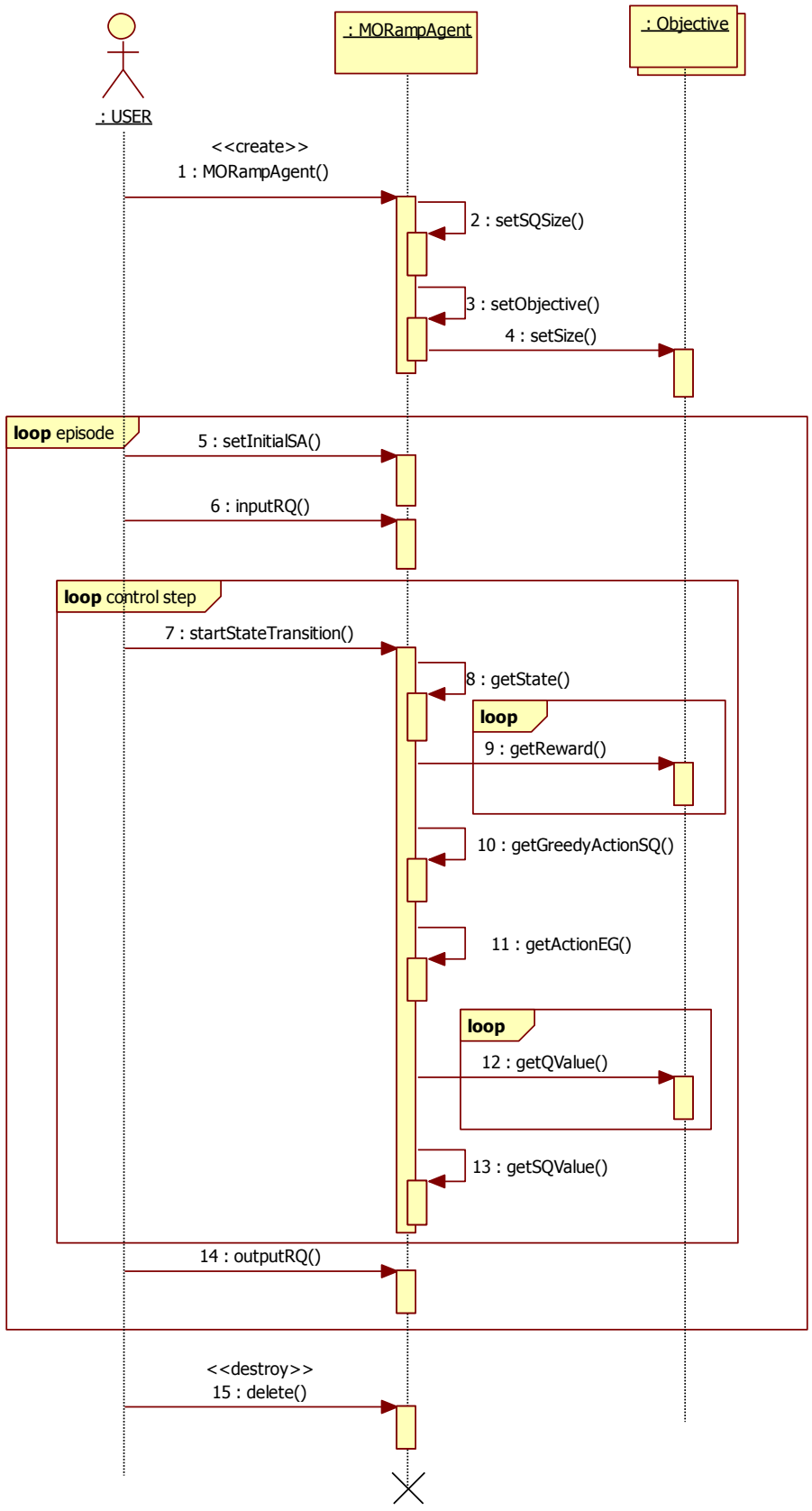


Figure 5.5: Sequence diagram for the multi-objective mode

These two sequence diagrams present the specific implementation of two algorithms, namely single-objective and multi-objective algorithms as previously introduced in Section 4.4. The flow charts in Figures 4.13 and 4.14 have described the working mechanisms of these two algorithms at an abstract level. In this section, two sequence diagrams will show how different functions (introduced in Sections 5.2.2 and 5.2.3) can be called and used to realise two algorithms at an implementation level.

In the single-objective mode (Figure 5.4), the learning process starts with the initialisation of a ramp agent through the constructor ***SORampAgent()***. Then two loops related to each episode (including ***setInitialSA()***, ***inputRQ()***, ***startStateTransition()*** and ***outputRQ()***) and the control step are triggered. The main part of the learning process (state transition) is realised by ***startStateTransition()***, which contains all functions required for Q-learning such as ***getState()***, ***getReward()***, ***getGreedyAction()***, ***getActionEG()*** and ***getQValue()***. At the end of each learning process, ***delete()*** is used to release memory.

In the multi-objective mode (Figure 5.5), the constructor ***MORampAgent()*** instead of ***SORampAgent()*** is used to initialise the ramp agent. More than one control objective can be considered in this case, thus, besides two basic loops for episode and control step, two more loops for ***getReward()*** and ***getQValue()*** are maintained to deal with different objectives. In the state transition process of multi-objective case, the greedy action is selected by function ***getGreedyActionSQ()*** according to SQ value. Compared with single-objective case, one more function ***getSQValue()*** is required to scalarise Q values.

For real applications, two kinds of agents can be instantiated according to the number of control objectives required. The Q-learning process can be

conducted through calling related functions embed in the classes ***RampAgent***, ***SORampAgent***, ***MORampAgent*** and ***Objective***.

5.3 Summary

This chapter has introduced the detailed implementation issues related to ACTM and RAS. Both of these two parts were programmed by C++ and can be used as a platform to test RL algorithms under various traffic conditions (for different network layouts and different traffic demands).

The ACTM contains four kinds of cells expressed by four classes. For each kind of cell, two functions related to traffic flow dynamics and vehicle conservation can be called. In other works, these functions are software implementations of Equations (4.14) to (4.17) introduced in Section 4.2.

Two main modules of the ramp agent were implemented by two classes, namely the ***RampAgent*** class and the ***Objective*** class. Here, a special class ***Objective*** was developed to generalise the calculation process of different control objectives (regarding reward calculation and Q update). In this way, different objectives can be easily involved under the same framework. To deal with two different modes, the ***RampAgent*** class was extended to have two sub-classes, namely the ***SORampAgent*** class and the ***MORampAgent*** class. A number of functions embedded in these two classes can be called to complete the learning process and realise the single- and multi-objective algorithms introduced in Chapter 4.

For ACTM, all classes defined in Section 5.1 are contained in two files including the header file “trafficflowmodel.h” and implementation file “trafficflowmodel.cpp”. For a ramp agent, four files relating to the ***RampAgent*** module and the ***Objective*** module were developed. The ***RampAgent*** module contains two files “rampagent.h” and “rampagent.cpp”, while the ***Objective*** module is composed of the files “objective.h” and

“objective.cpp”. These six files contain the complete information regarding the declaration and implementation of developed classes for ACTM and RAS, which can be used together or separately according to different requirements. The full source code of these files can be found in Appendix A. The evaluation of RAS based on the platform developed in this chapter will be presented in Chapters 6 and 7.

CHAPTER 6 CASE STUDIES FOR HYPOTHETICAL NETWORKS

Following the description of the learning system and relevant software implementation, the following two chapters (6 and 7) will focus on case studies used to evaluate the proposed ramp agent system (RAS). To do the evaluation, a number of simulation experiments are designed and conducted in these two chapters using the platform developed in Chapter 5. Various aspects of RAS will be tested through three cases regarding two hypothetical networks (with single on-ramp and multiple on-ramps) and a real network selected from the M6 motorway in the UK. The performance of RAS is compared with the situation of no control and one of the most widely used control algorithms, ALINEA, in all three cases. Although a coordinated version of ALINEA, i.e. METALINE (as introduced in Section 2.3.3) can be considered in the multi-ramp case, it has been mentioned in (Papageorgiou and Kotsialos 2002) that METALINE had no advantages over ALINEA when recurrent congestion occurred (e.g. during daily peak hours). Since only recurrent congestion is considered in three cases, METALINEA will not be used for comparison.

This chapter will focus on the two hypothetical networks, while the real network will be discussed in Chapter 7. The aims of the evaluation of the two hypothetical cases are summarised as follows:

(1) In the single-ramp case, the aim is to use the simplest network with only one on-ramp to analyse the performance and characteristics of one ramp agent. This test is essential, because the whole RAS is based on the features of each ramp agent included. The analysis of one ramp agent can provide important information for extending from one agent to

a ramp agent system. The performance of one ramp agent will be tested regarding the influence of different learning parameters, the ability to improve traffic efficiency and the ability to deal with queue constraints.

(2) The multi-ramp case is an extension from the single-ramp case, which uses a network with three pairs of on- and off-ramps. The test here will focus on the performance of three ramp agents working as a system (RAS). Specifically, two abilities of the RAS will be tested, which comprises: the ability to improve traffic efficiency under different simplified demand profiles and the ability to maintain user equity in a scenario with an obvious inequity.

These two case studies are introduced in Sections 6.1 and 6.2 respectively. Section 6.3 gives the summary and discussions of this chapter.

6.1 Single-ramp Case

A simple network used by (Gomes and Horowitz 2003) is selected for the first case study. This network has shown its effectiveness for testing different control algorithms such as the capacity-demand strategy and ALINEA. In the single-ramp case, minimising TTS is the only objective considered, and without off-ramp, the only way of reducing TTS is to prevent the capacity drop phenomenon (the first mechanism introduced in Section 2.2.1).

6.1.1 Experiment design

Network layout

Figure 6.1 shows the layout of a single-ramp network which is a stretch of a typical three-lane motorway with one single-lane on-ramp. In ACTM, this network is divided into four cells including one on-ramp cell (cell 2), three normal cells (cells 0, 1 and 3). Thus, only one ramp agent is required to control the on-ramp cell 2.

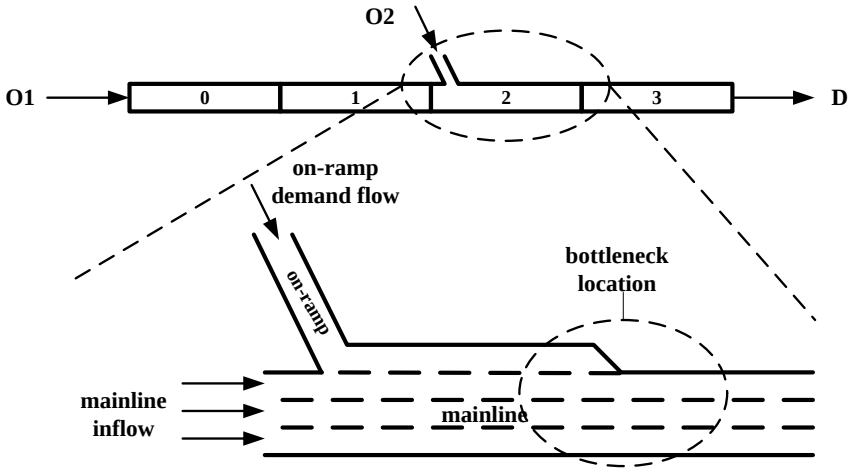


Figure 6.1: Network layout for the single-ramp case

Parameters for ACTM

The cell length and road capacity are selected from (Hegyi et al. 2005), where all cells have the same length l_i = 1 km, the capacity of each lane is set as 2000 veh/h, and thus for a three-lane mainline, $q_{cap,i}$ = 6000 veh/h. Following (Jiang and Chung 2013), the average vehicle length is assumed to be 4 metres and the minimum distance between two vehicles is 1 metre, then, each lane can contain up to 200 vehicles per kilometre. For a three-lane motorway, the jam density can be determined by: $\rho_{jam,i}$ = 600 veh/km (200 veh/lane/km). The free-flow speed v_i = 100 km/h and two specific parameters for ACTM including flow blending parameter θ_i = 0 and flow allocation parameter η_i = 0.16 are all selected from (Gomes and Horowitz 2006). According to these parameters, the congestion wave speed and critical density can be calculated as w_i = 11.1 km/h and $\rho_{crit,i}$ = 60 veh/km (20 veh/lane/km) respectively. T_s is set as 30 s to guarantee that $T_s \leq \min\{l_i / v_i\}$ (the CFL condition which was introduced in Section 4.2). A typical control interval T_c=30 s is selected (Papageorgiou et al. 2007), which is the same as the simulation interval T_s. It is assumed that a typical

capacity drop, i.e. 10% (Cassidy and Bertini 1999), will appear on the mainline when congestion occurs in the bottleneck location. Thus, λ = 0.9.

Demand profile

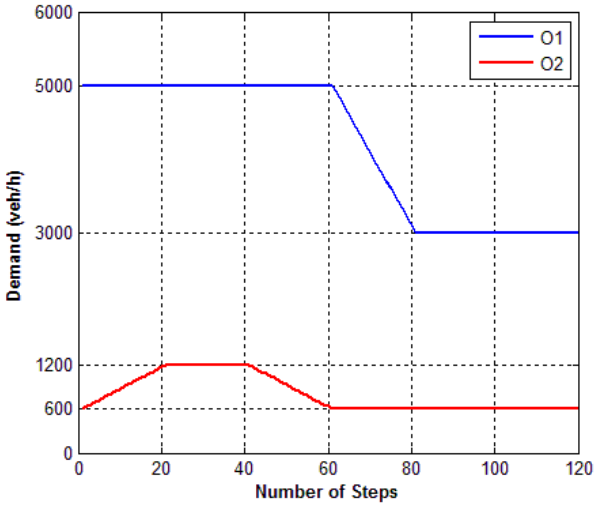


Figure 6.2: Demand profile for the single-ramp case

In the first hypothetical case, a trapezoidal demand profile for peak hours is adopted. This kind of demand profile simplifies the demand change in peak hours and provides a simple, but effective way to test the effectiveness of ramp metering. The trapezoidal demand profile has been used to test the performance of various ramp metering strategies in many simulation-based studies, such as model predictive control (Hegyi et al. 2005), ALINEA (Kotsialos et al. 2006), optimal control (Zhang and Shen 2010), and reinforcement learning (Davarynejad et al. 2011).

The trapezoidal demand profile in this study is presented in Figure 6.2 which is similar to the one used by (Hegyi et al., 2005) and (Davarynejad et al., 2011). The overall test period is 1 hour with 120 time steps (simulation interval T_s is 30 s). For the first 60 steps, a higher demand can cause traffic congestion on the motorway mainline. The decreased demand flow during the remaining 60 steps guarantees that traffic congestion can be completely

alleviated within the test period. Before each test period, there will be a warm-up period (60 steps) with a demand flow of 5000 veh/h for the mainline and 600 veh/h for the on-ramp. Under this setting, the system can reach a steady state by the beginning of the real test.

6.1.2 Strategy settings

In this chapter, all the simulation experiments are designed and carried out using ACTM. To combine RAS and ALINEA with ACTM, some necessary settings and modifications should be done here, as described below.

Modifications for ALINEA

The original ALINEA introduced in Section 2.3 was focused on the field applications which cannot be directly used in ACTM. Two modifications should be made to link ALINEA with ACTM.

(1) In the original ALINEA algorithm (Equation (2.16)), occupancy $o_{out,i}^{k}$ is used as the controlled variable, as it is a stable measurement that can be directly collected from loop detectors. Occupancy is defined as the proportion of time during which a detector is occupied by vehicles, and it can be converted to the density through some observed linear relationships (Kim and Hall 2004). Under the simulation environment of ACTM, occupancy cannot be directly generated. Thus, to combine ALINEA and ACTM, density ρ_i^k can be used to replace occupancy $o_{out,i}^{k}$ as mentioned in (Gomes and Horowitz 2003).

(2) Another problem is that the calculated metering rate may not be the same as measured on-ramp flow in real applications. To make a better calculation, these two rates should be distinguished by using the measured on-ramp flow $m_{r,i}^{k-1}$ from the last time step $k-1$ to update the calculated metering rate c_i^k at current step k (Papageorgiou et al. 1997).

Through the aforementioned modifications, the ALINEA updating equation used in ACTM is given below:

$$c_i^k = m_{r,i}^{k-1} + K_R(\hat{\rho}_i - \rho_i^k) \tag{6.1}$$

To apply ALINEA, two parameters regarding K_R (regulatory parameter) and $\hat{\rho}_i$ (target density) should be set. The calibration of these two parameters is shown in Appendix B.1 where two parameter values K_R = 0.3 and $\hat{\rho}_i$ = 20 veh/lane/km are found to be optimal.

In this study, two ALINEA-related algorithms namely ALINEA-C and ALINEA-D are used as a comparison. ALINEA-C is the theoretical application of ALINEA, by which the metering rates are directly calculated from Equation (6.1). Thus, continuous metering rates that may not be integer can be generated. The ALINEA-D (Kotsialos et al. 2006), on the other hand, is more practical, as it only allows an integer number of vehicles to enter the motorway mainline during each control interval. The discrete metering rates can be calculated by rounding the rate values generated by Equation (6.1) to the nearest integer numbers. By setting the same minimum and maximum metering rates, ALINEA-D can generate discrete metering rates in the same range of RAS, which guarantees a fair comparison.

Settings of RAS

For real applications, the action set and state set of RAS should be defined and regulated according to different network conditions. In a typical single-lane on-ramp used in this section, the minimum and maximum metering rates can be set as typical values: 240 veh/h and 1200 veh/h (Arnold Jr 1998). Then the set of discrete metering rates can be represented by $C_I = \{2,3,4,5,6,7,8,9,10\}$ veh/ T_c with 9 discrete metering rates that cover all possible values within the predefined range.

In Section 4.3.4, a four-dimensional state space containing four sub-sets $S_{nmain,I}$, $S_{qin,I}$, $S_{non,I}$ and $S_{don,I}$ has been defined for ramp metering. The main problem considered in this section is how to divide these four state sets. In this study, all state sets are uniformly divided into a number of states, and the more states there are, the more smooth the learning process is. However, an increased number of states for each state set will exponentially increase the size of state space, and thus increase the burden of computer memory and searching time in this state space. In this study, a suitable state deviation that enables a relatively smooth learning process with an acceptable number of states is as follows [6]:

(1) $\left|S_{nmain,I}\right|$=32 ($n_{main,i}^{up}$ = 600 veh, $n_{main,i}^{low}$=0 veh, $\Delta n_{main,i}$=20 veh)

(2) $\left|S_{qin,I}\right|$=12 ($q_{in,i}^{up}$ = 6000 veh/h, $q_{in,i}^{low}$ = 3000 veh/h, $\Delta q_{in,i}$=300 veh/h)

(3) $\left|S_{non,I}\right|$ = 12 ($n_{on,i}^{up}$ = 100 veh, $n_{on,i}^{low}$=0 veh, $\Delta n_{on,i}$=10 veh)

(4) $\left|S_{don,I}\right|$ = 12 ($d_{on,i}^{up}$ = 1200 veh/h, $d_{on,i}^{low}$=600 veh/h , $\Delta d_{on,i}$=60 veh/h)

where, $n_{main,i}^{low}$ and $n_{on,i}^{low}$ are determined according to the minimum possible number of vehicles on the mainline and on-ramp which are both 0, $n_{main,i}^{up}$ is the maximum possible vehicles on the mainline which can be calculated by $n_{main,i}^{up} = \rho_{jam} \cdot l_i$ = 600 veh, $n_{on,i}^{up}$ = 100 veh is the maximum acceptable vehicle queue length at on-ramp. $q_{in,i}^{up}$ = 6000 veh/h is the maximum inflow of a cell, i.e. the capacity flow. $q_{in,i}^{low}$ = 3000 veh/h, $d_{on,i}^{up}$ = 1200 veh/h and $d_{on,i}^{low}$ = 600 veh/h are all determined from the demand profile (Figure 6.2) which

---

6 The main system configuration of the computer in this study is as follows: the CPU is an Intel Core 5 CPU, 1.18 GHz and the installed memory is 2.92 GB. It is found in this study that when the number of states of one ramp agent is around 60000, the learning process is relatively smooth and does not take too much time (usually within one hour). For other computers with different configurations, the acceptable state number may be different.

indicates the highest and lowest flows from both the mainline and on-ramp. Thus, the total number of states is $|S_I|$ = 32×12×12×12 = 55296 (can be calculated from Equation (4.38)).

Besides the action and state, three learning parameters, namely learning rate α, action selection parameter ε, and discount rate γ should be well set to guarantee a good performance of RAS. To find suitable parameter settings, these three parameters will be analysed in Section 6.1.3.

6.1.3 Learning parameters analysis

A simple sensitivity analysis named One-at-a-Time (OAT) (Saltelli 1999) is used here to find suitable parameter settings and analyse the influence of different learning parameters. The minimum TTS of cell 2 achieved by RAS is around 3920 veh.min (this value can be obtained by the test shown in Section 6.1.4), which will be set as a benchmark in OAT analysis. Once the ramp agent reaches this benchmark line, it has learned the required control actions. The influence of three parameters on the algorithm performance is tested according to two aspects: learning speed and convergence stability. The number of episodes (NE) spent to reach the benchmark line is used as an indicator of learning speed. The higher NE is, the slower the agent learns to find the required result. The convergence stability is measured by the variance of results (VR) after the benchmark has been reached. Higher VR means lower stability.

The OAT analysis is conducted by regulating one parameter at a time, while keeping others fixed. For instance, if α is the parameter analysed, γ and ε will be set as their baseline values for the whole test period. Then, the parameter α will be changed slightly from its baseline value to observe the changes of NE and VR. Two sensitivity indices $\mathrm{SI(NS)} = |\Delta \mathrm{NE} / \Delta\alpha|$ and $\mathrm{SI(VR)} = |\Delta \mathrm{VR} / \Delta\alpha|$ are used to measure the changes of NE and VR

respectively. For γ and ε, the same method can be used to test their influence. A commonly used value 0.8 (Rezaee et al. 2012) is chosen as the baseline of γ. The other two baselines for α and ε are set as 0.05 and 0.01 which are their minimum values in the test.

Based on this method and the experimental design shown in Section 6.1.1, a series of experiments are conducted in this section. Each experiment runs for one million episodes taking about 25 minutes to guarantee the convergence (15 seconds for each 10000 episodes). The test results are discussed as follows.

Learning rate

Table 6.1: NE and VR for different learning rates

α	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50
NE ($\times 10^4$)	82.00	50.00	32.00	22.00	18.00	16.00	15.00	15.00	15.00	15.00
SI(NE) ($\times 10^4$)	—	640.00	360.00	200.00	80.00	40.00	20.00	0.00	0.00	0.00
$\overline{\text{SI}}$(NE) ($\times 10^4$)	148.9									
VR ($\times 10^4$)	0.75	1.84	2.09	4.93	13.04	24.63	30.07	41.73	45.62	57.05
SI(VR) ($\times 10^4$)	—	21.80	5.00	56.80	162.20	231.80	108.80	233.20	77.80	228.60
$\overline{\text{SI}}$(VR) ($\times 10^4$)	125.1									

Table 6.1 presents NE and VR corresponding to different learning rates, and Figure 6.3 shows examples (with α = 0.05, 0.15, 0.3 and 0.5) of TTS convergence. For cases with $\alpha > 0.5$, the algorithm performance is very unstable, which is not shown here. From the test results, it can be seen that the learning speed is very sensitive to α when it is less than 0.2. For learning rates greater than 0.2, the learning speed cannot be increased too much by increasing α, and the number of episodes spent keeps at the same level around 150000 episodes. The convergence stability, on the other hand, continues to decrease with the growth of α. Therefore, α is

suggested to be set close to 0.2 to avoid low stability and keep a relatively high learning speed.

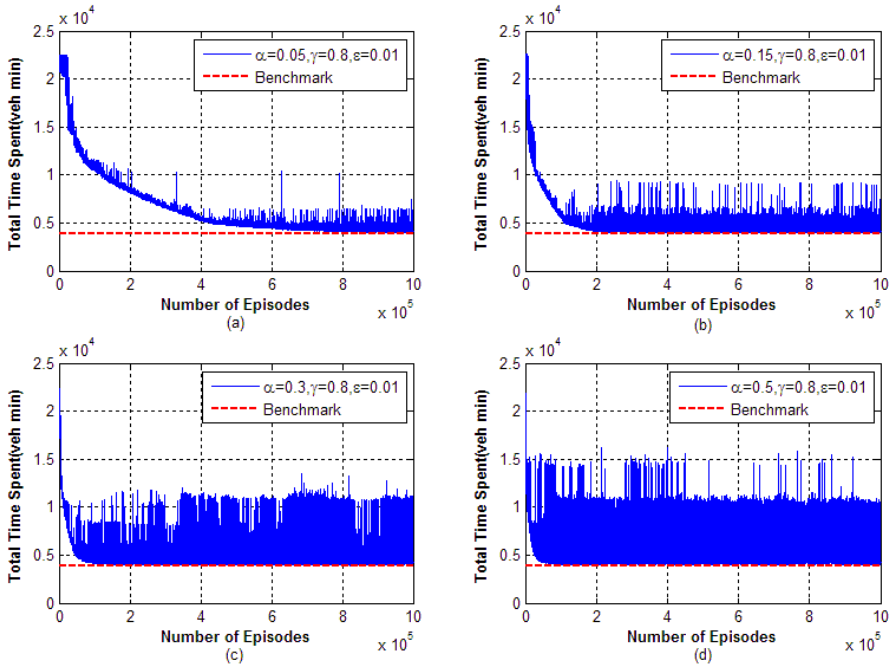


Figure 6.3: TTS convergence for different learning rates

Discount rate

Table 6.2: NE and VR for different discount rates

γ	0.50	0.55	0.60	0.65	0.70	0.72	0.75	0.78	0.80	0.85	0.90	0.95
NE ($\times 10^4$)	78.00	80.00	80.00	80.00	82.00	82.00	82.00	82.00	82.00	85.00	95.0	>100
SI(NE) ($\times 10^4$)	—	40.00	0.00	0.00	40.00	0.00	0.00	0.00	0.00	60.00	200.0	—
$\overline{\text{SI}}$(NE) ($\times 10^4$)	34.0											
VR ($\times 10^4$)	2.34	1.22	0.98	0.79	0.78	0.76	**0.73**	0.74	0.75	0.93	1.87	—
SI(VR) ($\times 10^4$)	—	22.40	4.80	3.80	0.20	0.40	1.00	0.20	0.40	3.60	18.80	—
$\overline{\text{SI}}$(VR) ($\times 10^4$)	5.6											

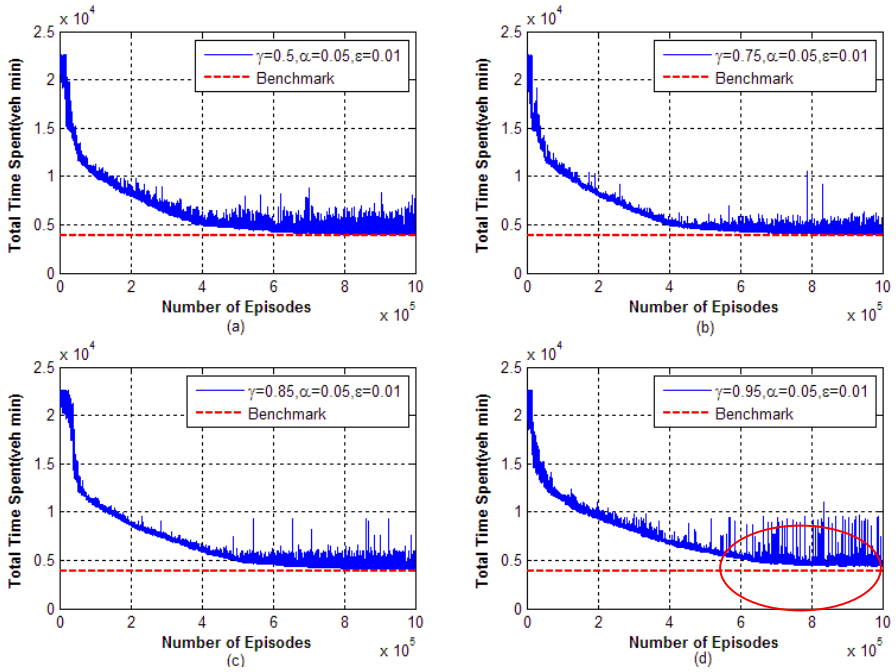


Figure 6.4: TTS convergence for different discount rates

With an increasing value of γ, the number of episodes required to approach the benchmark grows gradually (see Table 6.2). When γ reaches 0.95 (as shown in Figure 6.4 (d)), the benchmark value cannot even be achieved within one million episodes. For the stability test, one interesting finding is that VR does not fall all the time with the growth of γ. Indeed, one flexion point arises between 0.7 and 0.8 (in this test, this point is 0.75), around which the highest stability can be obtained. From the test of γ, it can be concluded that the learning speed is not sensitive to the discount rate, and the highest γ cannot guarantee the best stability. Thus, a value between 0.7 and 0.8 (0.75 in this test) should be chosen for γ to achieve the highest stability.

Action selection parameter

Table 6.3: NE and VR for different action selection parameters

ε	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10
NE ($\times 10^4$)	82.00	58.00	42.00	40.00	34.00	32.00	32.00	30.00	30.00	30.00
SI(NE) ($\times 10^4$)	—	2400.00	1600.00	200.00	600.00	200.00	0.00	200.00	0.00	0.00
$\overline{\text{SI}}$(NE) ($\times 10^4$)	577.8									
VR ($\times 10^4$)	0.75	2.47	7.76	11.55	22.16	43.74	66.28	79.70	93.78	94.26
SI(VR) ($\times 10^4$)	—	172.00	529.00	379.00	1061.00	2158.00	2254.00	1342.00	1408.00	48.00
$\overline{\text{SI}}$(VR) ($\times 10^4$)	1039.0									

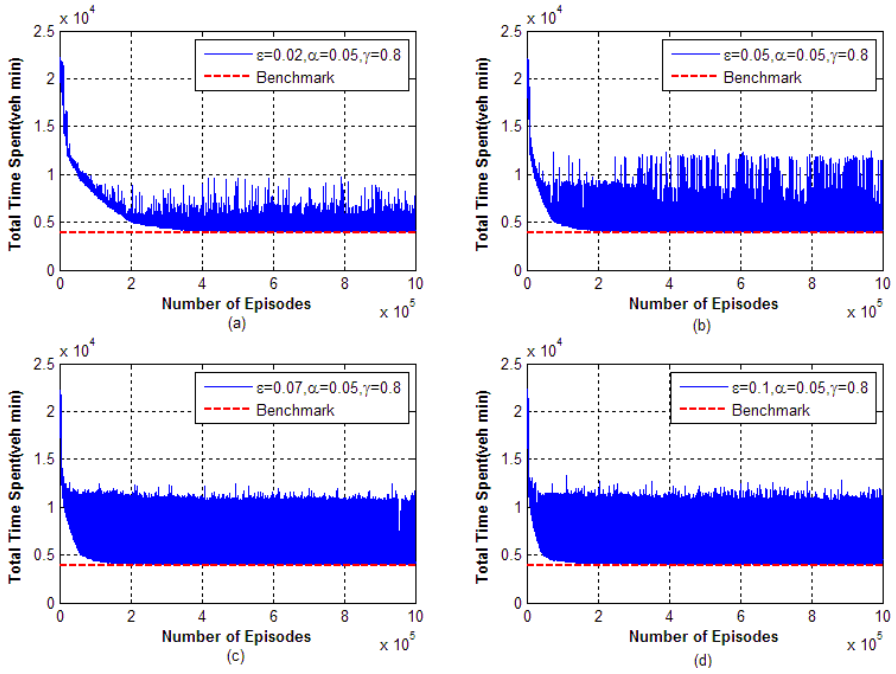


Figure 6.5: TTS convergence for different action selection parameters

From Table 6.3, it can be seen that both NE and VR are very sensitive to ε (examples of TTS convergence of ε = 0.02, ε = 0.05, ε = 0.07, ε = 0.1 are shown in Figure 6.5). Higher ε leads to lower stability. When ε reaches 0.1, the VR is already 94.26 ($\times 10^4$). On the other hand, NE reduces with the growth of ε, while after ε = 0.05, the learning speed cannot be improved

greatly with NE around 30 ($\times 10^4$). Therefore, it is better to set ε as a very small value such as 0.01 to obtain acceptable convergence stability.

In summary, the most sensitive parameter for both learning speed and convergence stability is ε with the highest average indices 577.8 ($\times 10^4$) and 1039.0 ($\times 10^4$). α has less impacts on the algorithm than ε and it has the average indices 148.9 ($\times 10^4$) for learning speed and 125.1 ($\times 10^4$) for convergence stability. Compared with α and ε, the discount rate γ seems to be less important with average sensitivity indices 34.0 ($\times 10^4$) and 5.6 ($\times 10^4$) for the learning speed and convergence stability, respectively.

Through the comparison of different parameter values, one possible parameter setting is given as: α = 0.2, γ = 0.75, ε = 0.01 that can guarantee a high learning speed without losing too much stability. These parameters will be used for the remaining tests in both Chapters 6 and 7.

6.1.4 Efficiency test

As mentioned earlier, in the single-ramp case, efficiency improvement is the only objective considered. In the first case study, an efficiency test is conducted by comparing RAS with a non-controlled situation (NC) and ALINEA.

Non-controlled situation

As shown in Figure 6.6, without control, during the first 60 time steps, because of the high demand from both mainline and on-ramp, congestion occurs in cell 2 (between 2 and 3 km) and propagates upstream to cell 1 (between 1 and 2 km). The most severe congestion occurs around the 50th time step with the highest density in cell 2 around 70veh/lane/km. For the last 60 time steps, the reduction of traffic demand alleviates traffic

congestion, and after nearly 100 steps, traffic flow returns to free-flow state. During the whole test period of the non-controlled situation, the TTS of cell 2 is 7160 veh.min, while the network TTS reaches 17156 veh.min.

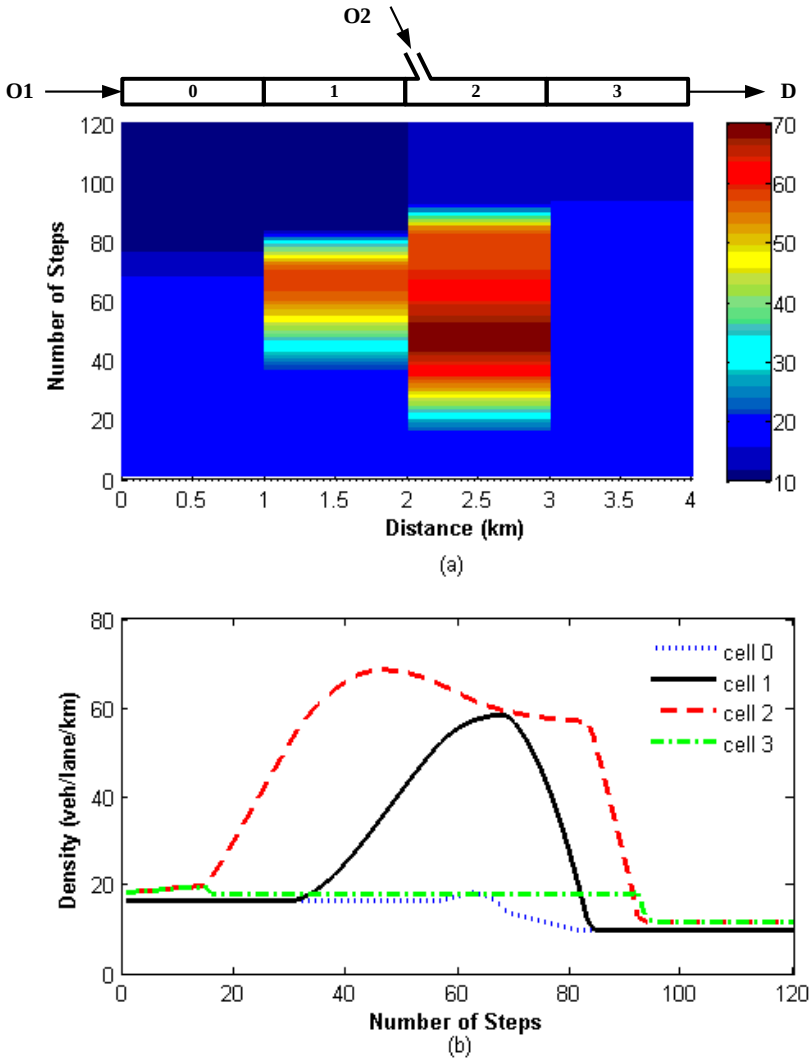


Figure 6.6: Density without control: (a) density evolution, (b) cell density

Controlled situation

Figure 6.7 illustrates the convergence of RAS (shows the TTS of cell 2) under parameter settings selected from Sections 6.1.2 and 6.1.3. The optimal control actions can be found after 220000 episodes which take about 5 minutes.

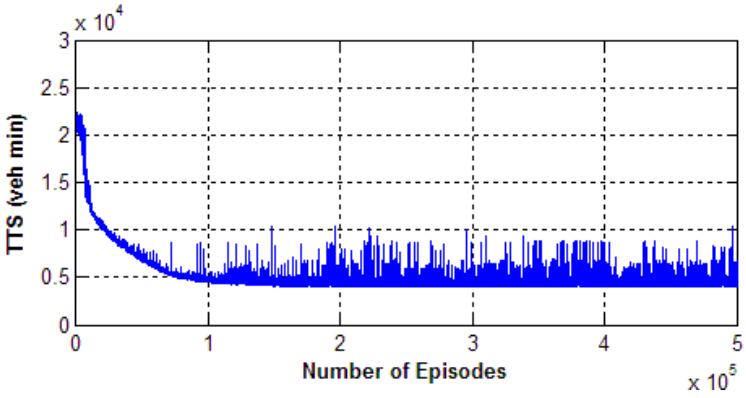


Figure 6.7: TTS convergence of RAS

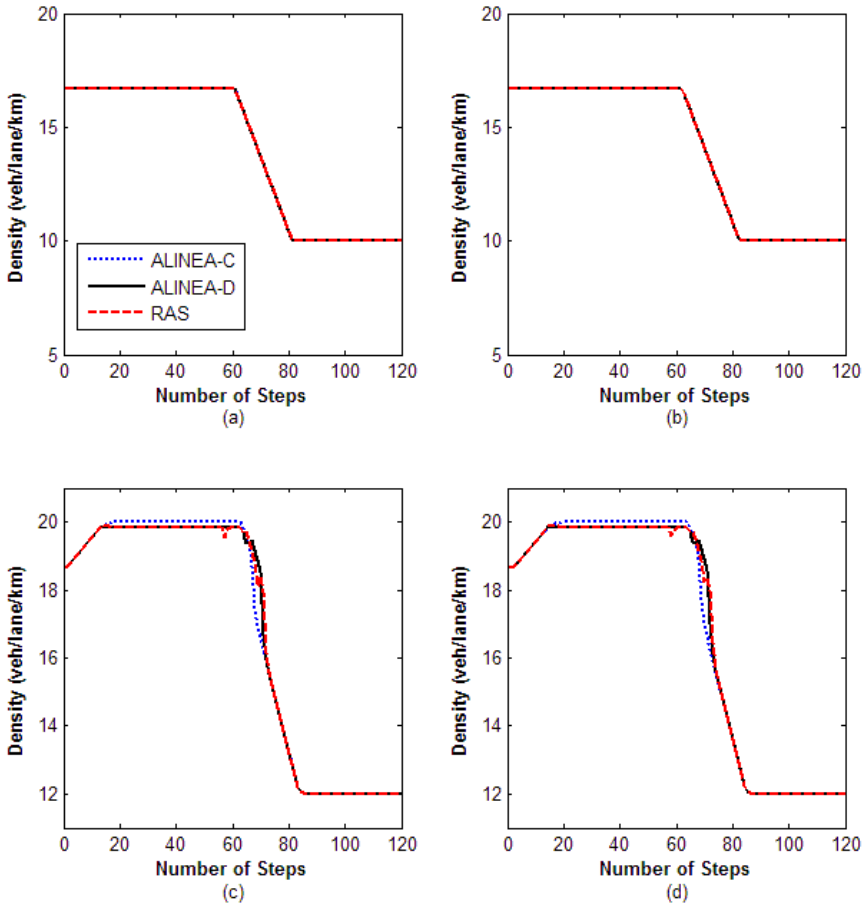


Figure 6.8: Cell density with control: (a) cell 0, (b) cell 1, (c) cell 2, (d) cell3

As shown in Figure 6.8, all three strategies can eliminate network congestion by keeping mainline density below the critical value. For TTS reduction, RAS and ALINEA-D have almost the same performance (see Figure 6.9). Under

these two control strategies, as shown in Table 6.4, the TTS of cell 2 during the whole test period can be reduced by 45.3%, and the network TTS can be reduced by 30.2%. ALINEA-C is better than RAS and ALINEA-D, which can reduce the TTS of cell 2 by 48.8% and for the network by 31.7%.

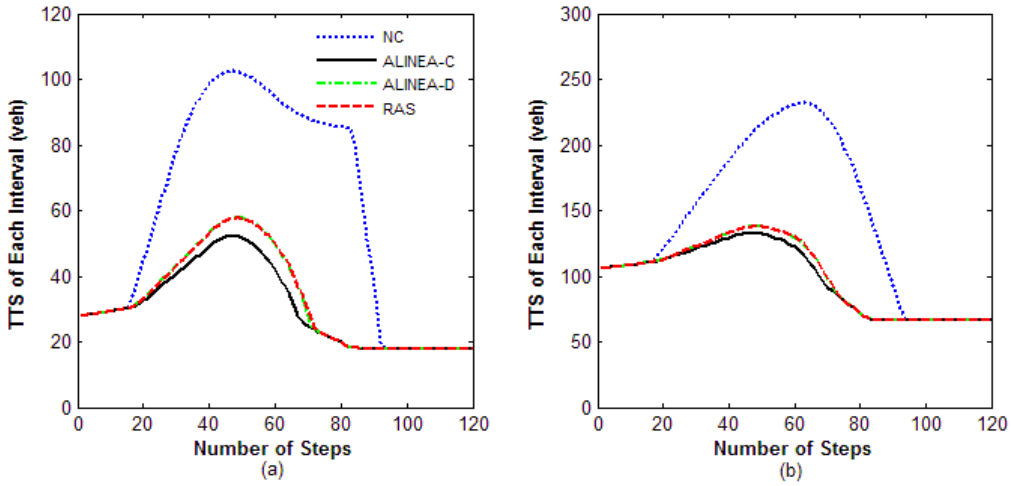


Figure 6.9: TTS comparison for: (a) cell 2, (b) network

Table 6.4: TTS comparison

Strategies	TTS of cell 2 (veh.min)	Reduction (%)	TTS of the network (veh.min)	Reduction (%)
NC	7160	_	17156	_
ALINEA-C	3664	48.8	11725	31.7
ALINEA-D	3917	45.3	11971	30.2
RAS	3919	45.3	11977	30.2

The main reason for this result can be explained by Figure 6.8 (c), from which it can be seen that ALINEA-C can make the mainline density exactly the same as the critical value (20 veh/lane/km) between time steps 10 and 80 by using continuous metering rates. In this situation, the outflow of cell 2 can keep at a higher level, which leads to a shorter queue length as shown in Figure 6.10, and thus has lower TTS. On the other hand, the critical density cannot be strictly reached by RAS and ALINEA-D with discrete

integer metering rates. Thus, these two strategies have a longer queue length and higher TTS than ALINEA-C.

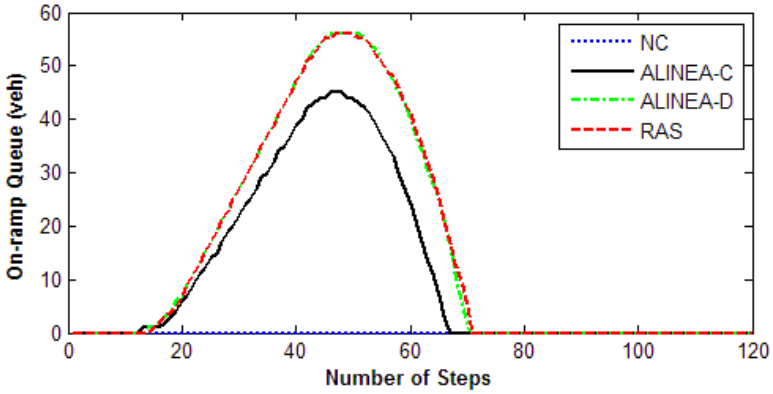


Figure 6.10: On-ramp queue comparison

6.1.5 Queue constraints test

It has been mentioned in Section 2.3 that a successful ramp metering strategy should be able to keep the on-ramp queue within a predefined constraint to avoid interrupting the traffic operation in local streets. A popular way to tackle queue constraints is to combine existing metering strategies with a queue management algorithm, such as ALINEA/Q introduced in Section 2.3.2 (Equations (2.17), (2.18) and (2.19)). RAS developed in this study, on the other hand, does not need an extra queue management algorithm. In RAS, different queue constraints can be considered through setting different queue boundaries, i.e. $n_{on,i}^{\max}$ in Equation (4.31).

In this section, the ability of RAS to deal with queue constraints is tested and compared with ALINEA/Q. Here, ALINEA/Q consists of two algorithms namely ALINEA-C/Q and ALINEA-D/Q corresponding to ALINEA-C and ALINEA-D respectively. Queue estimation given by Equation (2.17) is not required in our test, as the exact on-ramp queue generated by ACTM can be directly obtained. Five different queue constraints (10 veh, 20 veh, 30 veh,

40 veh, 50 veh) are used to form the test. The simulation results are shown below.

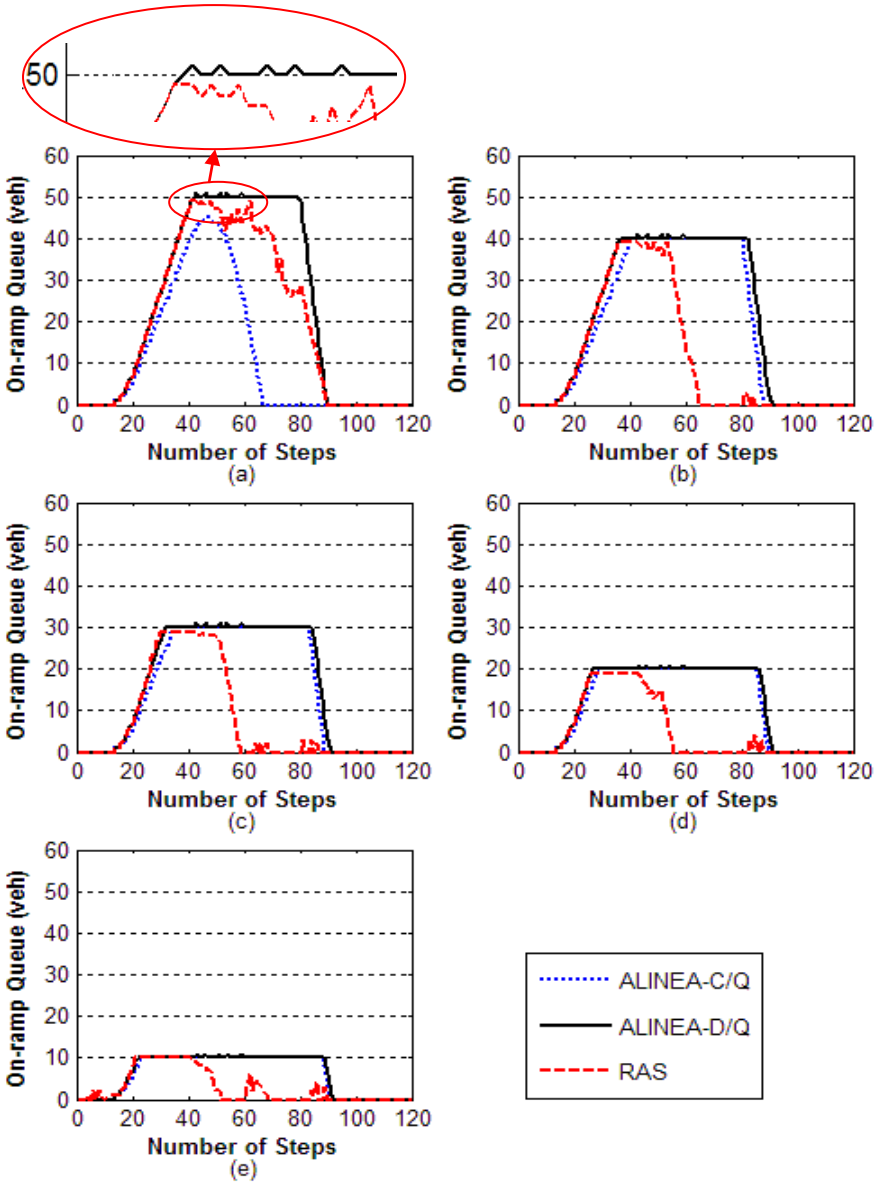


Figure 6.11: On-ramp queue comparison under queue constraints: (a) 50 veh, (b) 40 veh, (c) 30 veh, (d) 20 veh, (e) 10 veh

Figure 6.11 shows on-ramp queues under different control strategies and queue constraints. RAS can successfully keep the on-ramp queue under constraint in all five situations. By taking continuous metering rates, ALINEA-C/Q can also restrict the ramp queue length under predefined constraints

during the whole control period. With discrete metering rates, ALINEA-D/Q cannot work as well as ALINEA-C/Q. At some time steps, the on-ramp queue may exceed its maximum acceptable value under the control of ALINEA-D/Q (as shown in Figure 6.11 (a)). This is mainly because the metering rate generated by ALINEA-D/Q needs to be rounded to the nearest integer number.

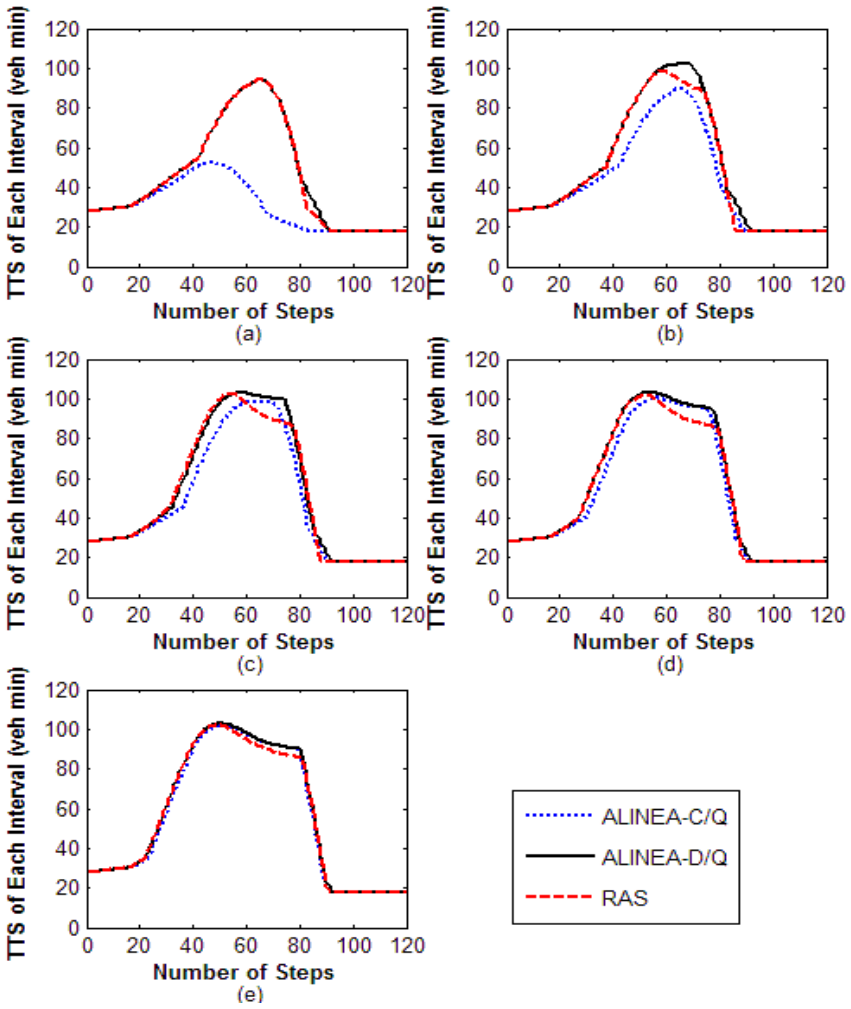


Figure 6.12: Cell 2 TTS comparison under queue constraints: (a) 50 veh, (b) 40 veh, (c) 30 veh, (d) 20 veh, (e) 10 veh

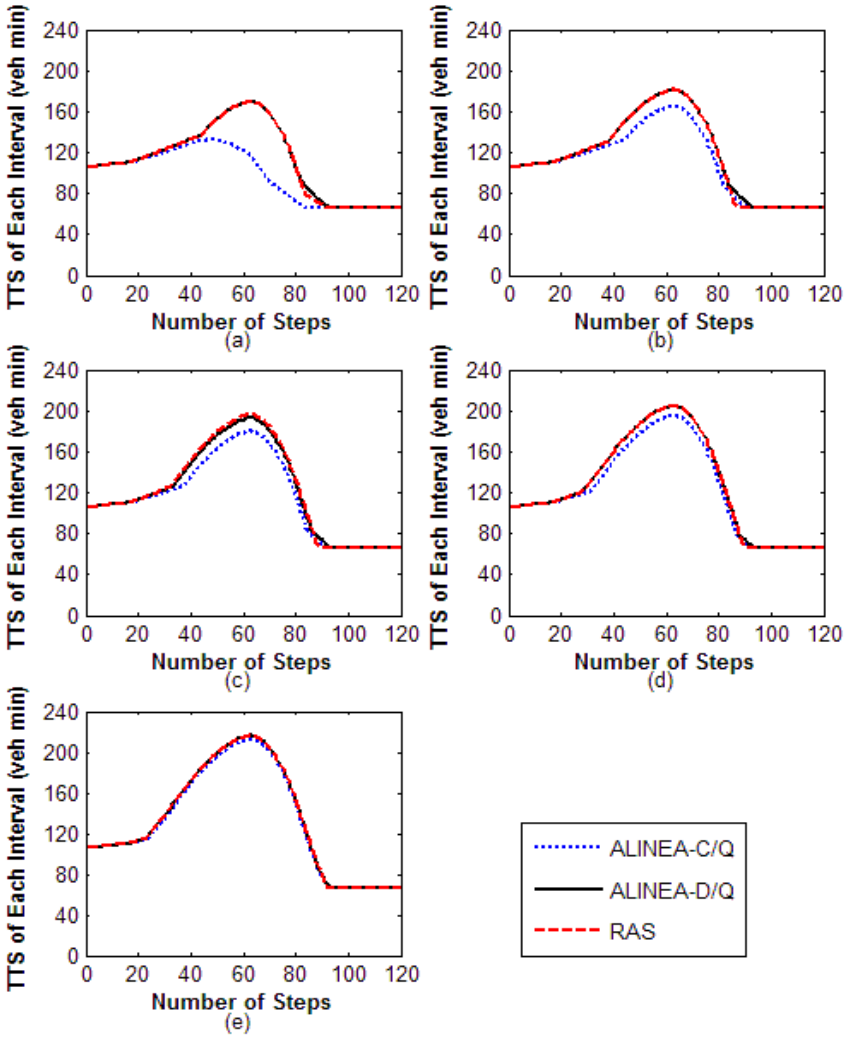


Figure 6.13: Network TTS comparison under different queue constraints: (a) 50 veh, (b) 40 veh, (c) 30 veh, (d) 20 veh, (e) 10 veh

For the TTS comparison illustrated in Figures 6.12 and 6.13, ALINEA-C/Q has lower TTS than RAS and ALINEA-D/Q under all five queue constraints, because it can release a continuous number of vehicles. With the same range of discrete metering rates, RAS can reduce more TTS of cell 2 than ALINEA-D/Q (Figure 6.12), while RAS and ALINEA-D/Q possess a similar performance on reducing TTS for the whole network (Figure 6.13). This result can be explained by an example shown in Figure 6.14 where densities under different control strategies with queue constraint of 30 veh are compared.

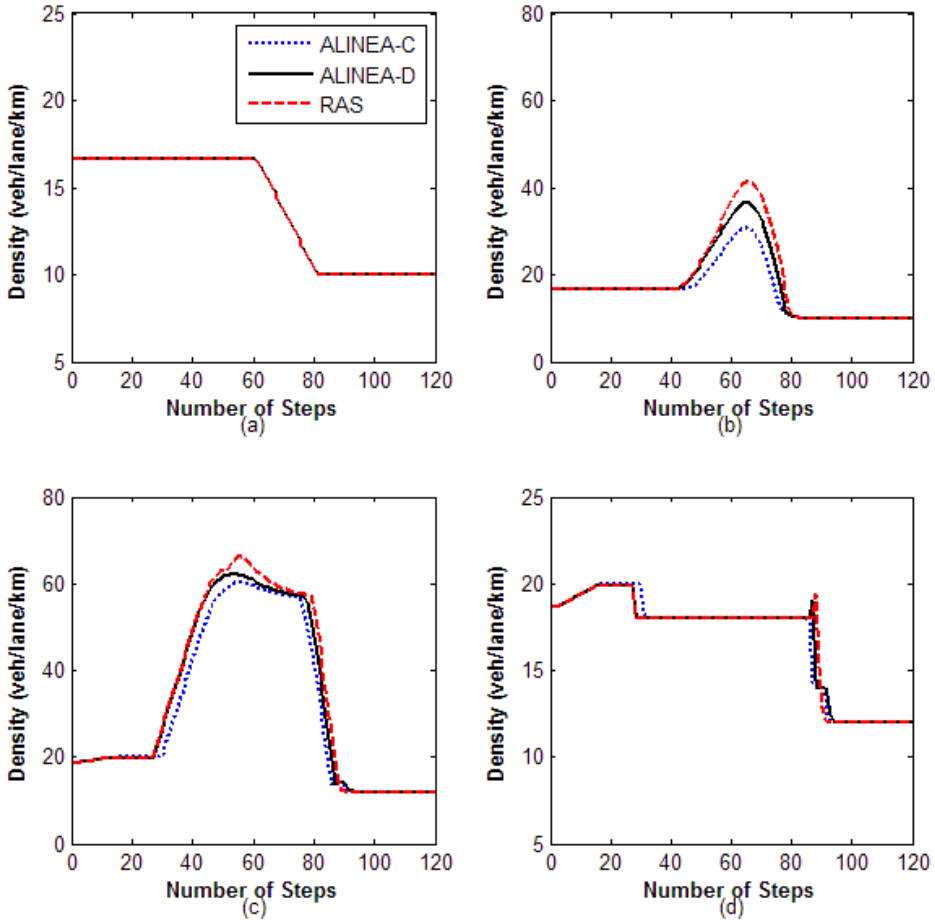


Figure 6.14: Cell density with queue constraint 30 veh: (a) cell 0, (b) cell 1, (c) cell 2, (d) cell 3

Because of the queue constraints of on-ramp, extra vehicles that exceed the constraint should be released to the motorway mainline. In this situation, the mainline density cannot be maintained around the critical value, and congestion cannot be eliminated (see Figure 6.14 (c)). Thus, the outflow of cell 2 cannot be improved and will stay at a lower value equalling the queue discharge rate (because of capacity drop). In this test, the main objective of RAS is to minimise the TTS of cell 2. Compared with ALINEA, RAS is less strict on maintaining on-ramp queue length and lets more vehicles stay on the mainline. Therefore, fewer vehicles can be received by cell 2 which leads to lower inflow to cell 2. As the outflow does not change, the reduced inflow of cell 2 can reduce the TTS of this cell. However, as shown in Figure 6.14

(b), the reduced inflow to cell 2 causes more severe congestion in its upstream cell 1, which increases the TTS of cell 1. In this situation, there is little difference between the sum of these two TTSs (cell 1 and 2) controlled by RAS and ALINEA-D/Q. Moreover, as shown in Figure 6.14 (a) and (d), the densities of cell 0 and 3 have no obvious changes using different control strategies. Therefore, the network TTS (the sum of TTS of cell 0, 1, 2 and 3) controlled by RAS is close to the one controlled by ALINEA-D/Q.

6.1.6 Summary

In this section, a number of simulation experiments were conducted to analyse and test the performance of RAS comprising only one ramp agent.

Through the analysis of learning parameters, it was found that the learning speed is very sensitive to α when it is less than 0.2, while the most sensitive parameter is ε which should be set as a very small value. Compared with α and ε, γ had the least effect on both learning speed and convergence stability. Based on these findings, a group of parameter settings with α = 0.2, γ = 0.75, ε = 0.01 that can balance the speed and stability was selected and used for the following tests.

In the efficiency test, RAS showed a good performance on improving traffic efficiency, which can reduce the network TTS by 30.2% from the non-controlled situation. This performance was almost the same as ALINEA-D, but not as good as ALINEA-C which can reduce TTS by 31.7%.

In the queue constraints test, RAS demonstrated its ability to manage on-ramp queues under predefined constraints. Different from ALINEA which needs to be combined with a queue management algorithm, RAS can manage queue length by directly setting boundaries for its reward (i.e. values for $n_{on,i}^{\max}$). When the same range of discrete metering rates was used,

RAS had a lower TTS in the controlled cell than ALINEA, but the network TTS of the two strategies was similar.

6.2 Multi-ramp Case

The single-ramp case discussed in Section 6.1 analysed the performance of one ramp agent. This section will focus on the ramp agent system that contains more than one ramp agent. In the efficiency test shown in Sections 6.2.2, 6.2.3 and 6.2.4, improving traffic efficiency is the only objective. In the equity test presented in Section 6.2.5, except for efficiency improvement, maintaining user equity will be considered as an additional objective. Three demand profiles leading to different congestion levels will be used to form a series of simulation-based experiments.

6.2.1 Experimental design

Network layout

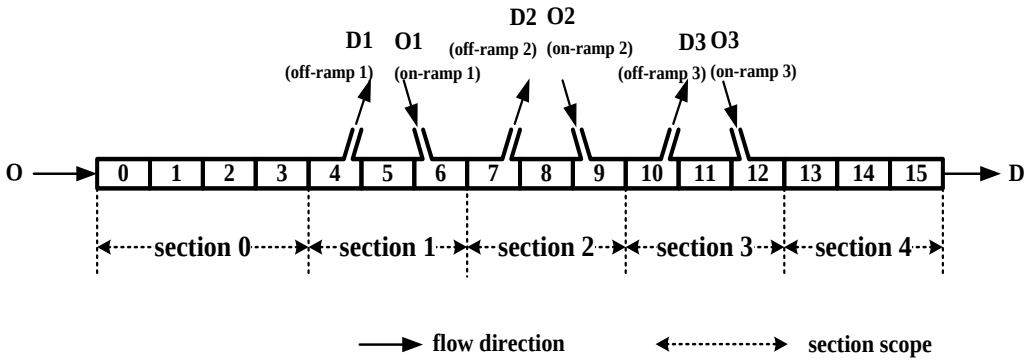


Figure 6.15: Network layout for the multi-ramp case

In the multi-ramp case, a motorway network with 16 cells is used here. This network is extended from the simple network described in Section 6.1. For ease of comparison, these cells are grouped into 5 motorway sections from section 0 to section 4, and one motorway section may contain more than one cell. Motorway sections 1, 2 and 3 are controlled sections including on-ramp cells 6, 9 and 12. Except for one on-ramp cell, each controlled section

contains one normal cell and one off-ramp cell as presented in Figure 6.15. For RAS, three ramp agents with index 1, 2 and 3 are used to control their corresponding on-ramps, i.e. on-ramp 1, 2 and 3. For ALINEA, three controllers are used, which have the same indices as their controlled on-ramps.

Parameters for ACTM

In the multi-ramp case, all cell lengths are set as 500 metres, and split ratios ($\beta_1, \beta_2, \beta_3$) for off-ramps (1, 2 and 3) are set as a typical value 0.1. To satisfy the CFL condition, T_s is set as 15 s. All other parameters are set as the same values in the single-ramp case, which are summarised as follows: $\rho_{crit,i}$ = 60 veh/km (20 veh/lane/km), $\rho_{jam,i}$ = 600 veh/km (60 veh/lane/km), $q_{cap,i}$ = 6000veh/h, v_i = 100 km/h, w_i = 11.1 km/h, θ_i = 0, η_i = 0.16, λ = 0.9.

6.2.2 Efficiency test I

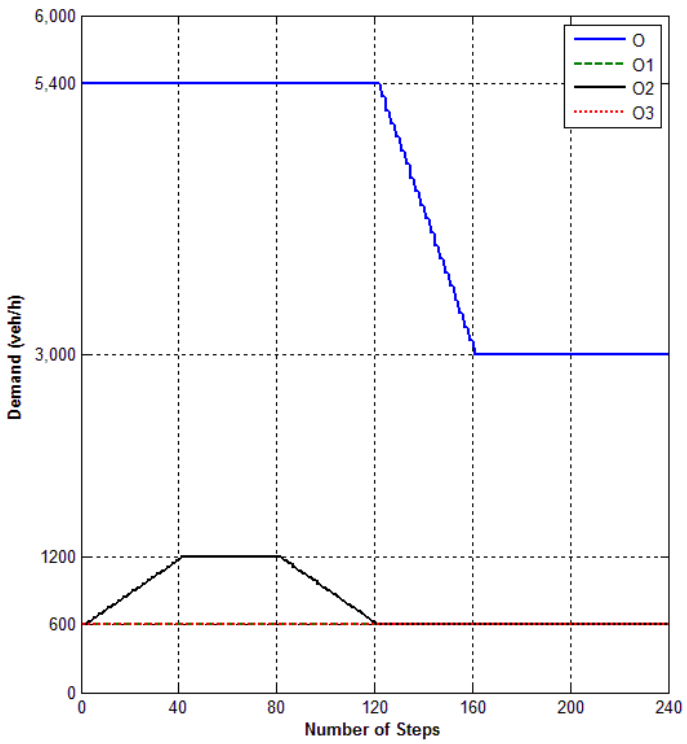


Figure 6.16: Demand profile 1

Figure 6.16 illustrates demand profile 1 where demand flows from on-ramp 1 (O1) and 2 (O2) will be kept at a low level around 600 veh/h during the whole test period, while the demand flow from on-ramp 3 (O3) will increase to 1200 veh/h during the first 120 time steps. Under this demand profile, congestion is caused by the demand flow from on-ramp 3 that has significant influence on motorway section 3.

Non-controlled situation

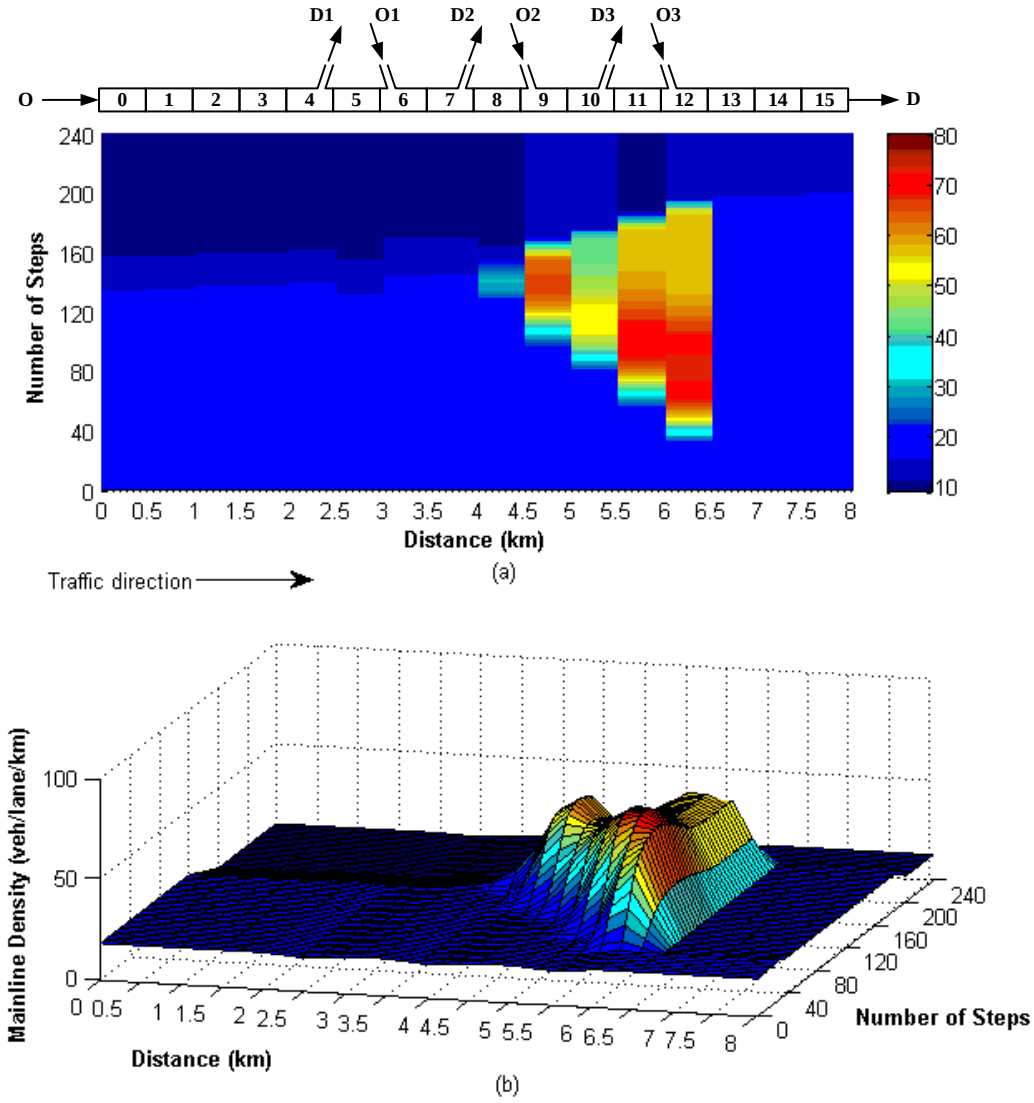


Figure 6.17: Density evolution under NC (demand 1)

Figure 6.17 shows the density evolution under the non-controlled situation (NC). Traffic congestion occurs in cell 12 (between 6 and 6.5 km) and propagates upstream to cell 8 (between 4 and 4.5 km) during the next 100

steps. With a lower demand starting from step 120, the congestion dissipates gradually. After 180 steps, no congestion can be found in the test network. During the whole test period without control, the network TTS is 28809 veh.min.

Controlled by RAS

Learning parameters of three ramp agents are set as the same values selected from the single-ramp case. These parameters worked well in all tests presented in Section 6.2, which will not be recalibrated in the following subsections. The same discrete metering rate $C_I = \{2,3,4,5,6,7,8,9,10\}$ veh/ T_c used in the single-ramp case will also be adopted by the multi-ramp case. As the cell length in the multi-ramp case is 500 metres, the maximum number of vehicles on the mainline should be $n_{main,i}^{up}$ = 300 veh (which is half of the single-ramp case), and the maximum on-ramp queue $n_{on,i}^{up}$ is set as 200 veh because of the heavier traffic load in the multi-ramp case. Other state-related parameters are all the same as the single-ramp case. Thus, the state number of each agent should be $|S_I|$ = 17×12×12×22= 53856. This state space will be used by all three ramp agents (controlling three on-ramps 1, 2 and 3) in all tests of the multi-ramp case.

The convergence of RAS with the selected parameters is shown in Figure 6.18, and it takes around 250000 episodes (15 minutes) to learn the optimal control actions. Figure 6.19 illustrates the density evolution under the control of RAS, from which it can be seen that congestion can be completely eliminated during the whole control period. When RAS is used, the network TTS can be reduced to 23195 veh.min, which is a 19.5% reduction compared with the non-controlled situation.

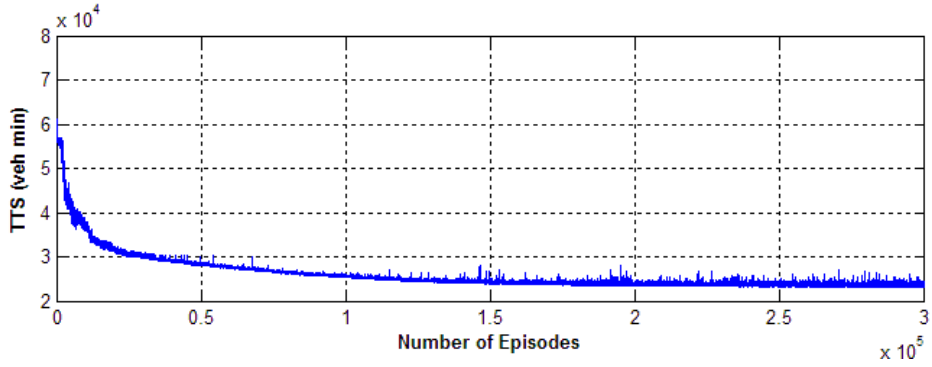


Figure 6.18: RAS convergence (demand 1)

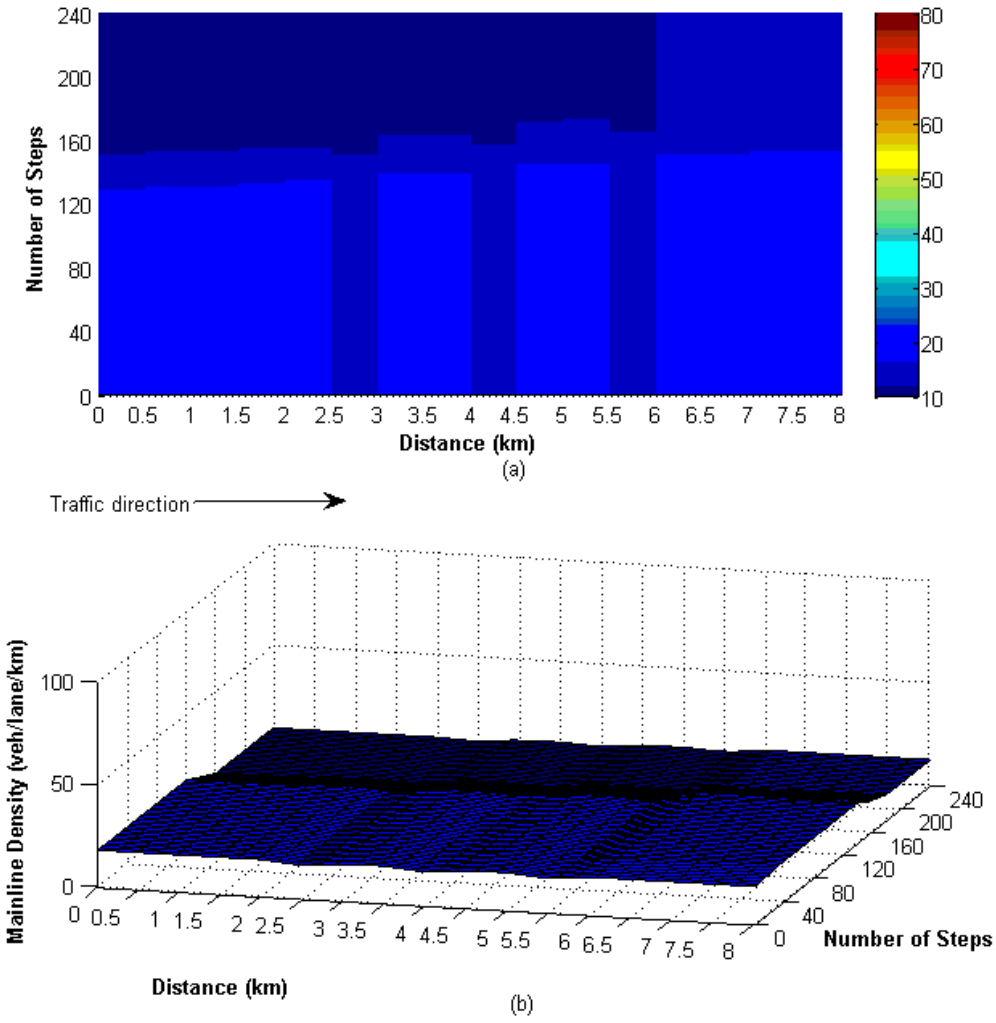


Figure 6.19: Density evolution under RAS (demand 1)

Comparison with ALINEA

The parameters of three ALINEA controllers (controlling three on-ramps) are also selected from the single-ramp case, which are the same for both ALINEA-C and ALINEA-D: $K_{R,1}=K_{R,2}=K_{R,3}$=0.3, $\hat{\rho}_1=\hat{\rho}_2=\hat{\rho}_3=\rho_{crit}$.

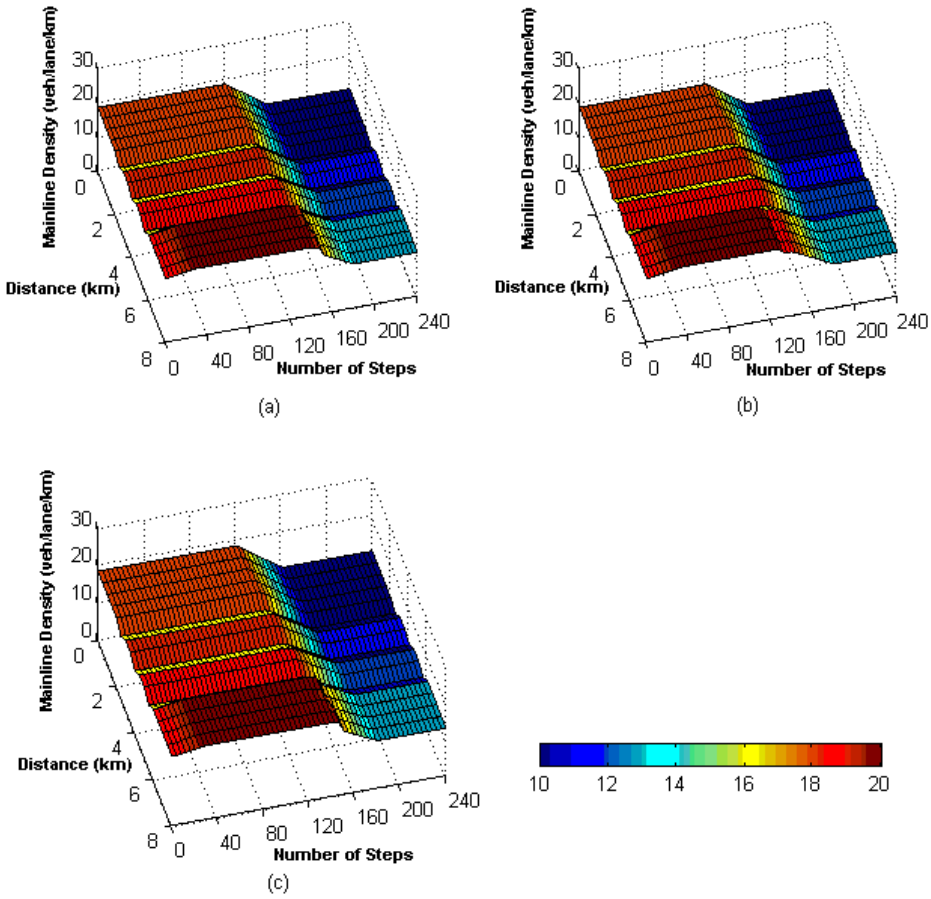


Figure 6.20: Density comparison (demand 1): (a) RAS, (b) ALINEA-C, (c) ALINEA-D

The density comparison of RAS, ALINEA-C and ALINEA-D is shown in Figure 6.20. All three strategies can keep a smooth density evolution below the critical value. Compared with RAS and ALINEA-D, lower TTS with a reduction of 21.0% can be obtained by ALINEA-C (see Table 6.5 and Figure 6.22). ALINEA-D provides almost the same performance as RAS which can reduce TTS by 19.5%. This result is similar to the single-ramp case, as only one on-ramp (on-ramp 3) may cause congestion and needs strict control.

Because no congestion can be caused by on-ramps 1 and 2, metering rates for these two on-ramps are both kept at the maximum value by the three strategies, which leads to no waiting vehicles on these two on-ramps (see Figure 6.21).

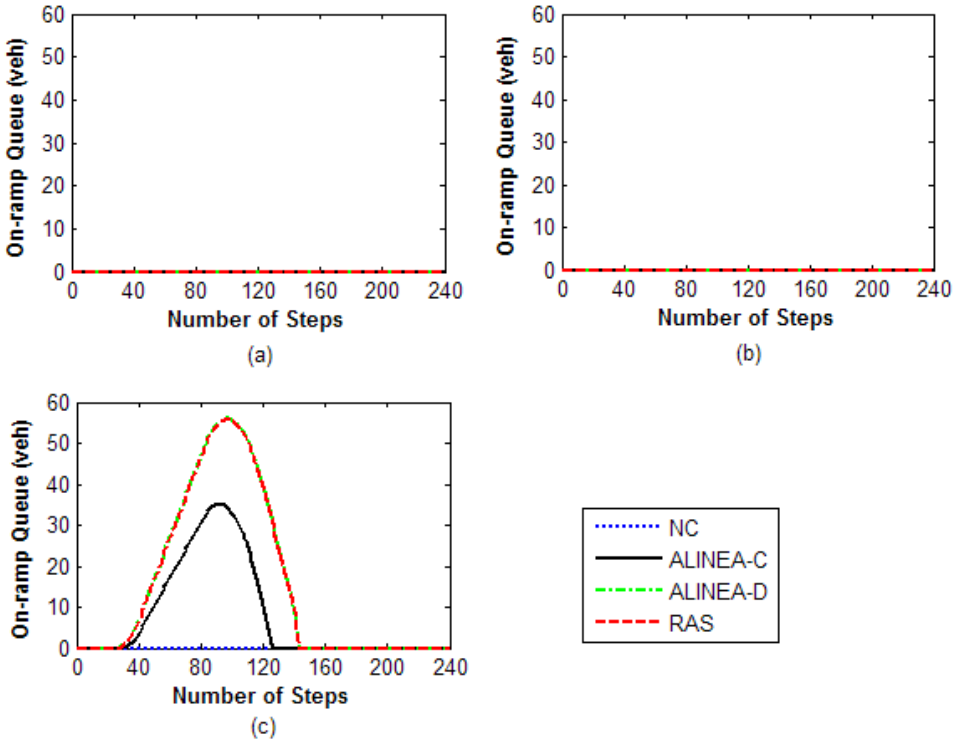


Figure 6.21: On-ramp queue comparison (demand 1) for: (a) section 1, (b) section 2, (c) section 3

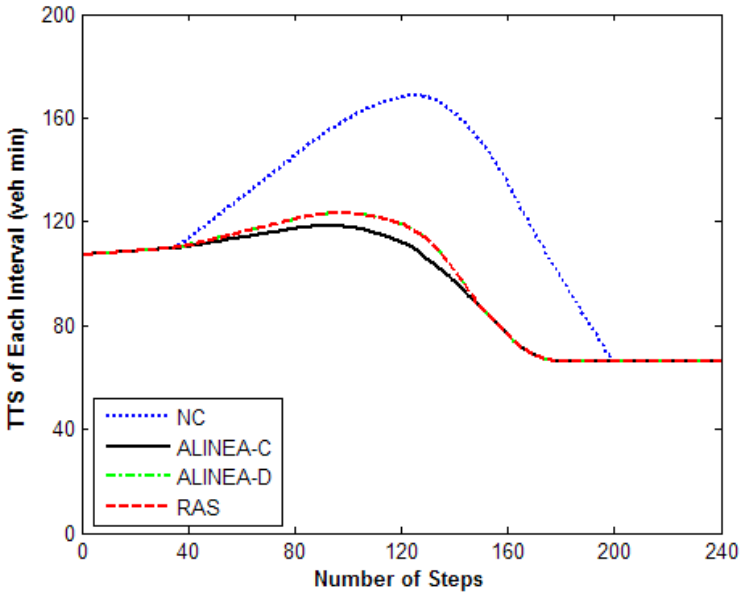


Figure 6.22: Network TTS comparison (demand 1)

Table 6.5: TTS comparison (demand 1)

Strategies	TTS for the whole period (veh.min)						
	Section 0	Section 1	Section 2	Section 3	Section 4	Network	Reduction (%)
NC	5316	3936	5143	9766	4648	28809	–
ALINEA-C	5316	3936	4092	4760	4648	22752	21.0
ALINEA-D	5316	3936	4092	5210	4648	23202	19.5
RAS	5316	3936	4092	5203	4648	23195	19.5

6.2.3 Efficiency test II

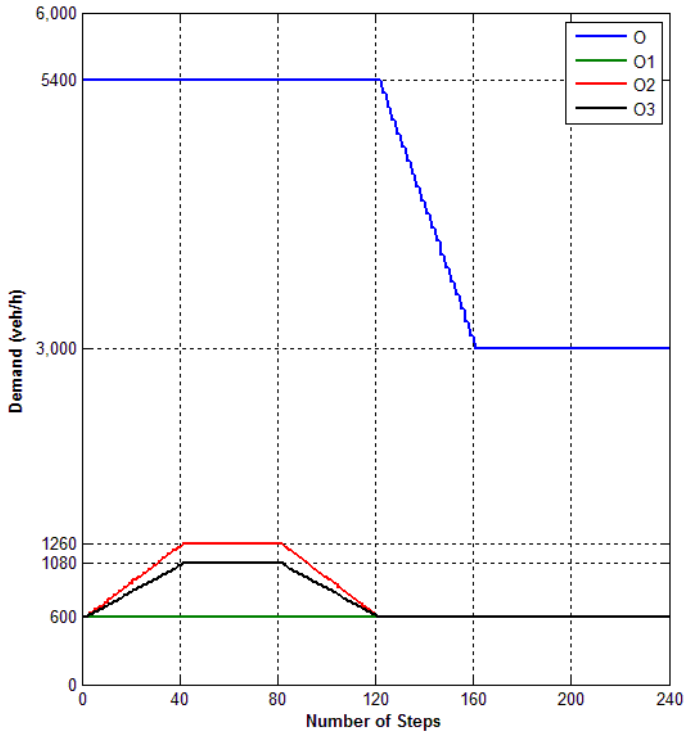


Figure 6.23: Demand profile 2

In the second demand situation shown in Figure 6.23, demand flows from the mainline (O) and on-ramp 1 (O1) will keep at the same values as in the first demand profile during the whole test period. For on-ramp 2 (O2) and 3 (O3), demand flows will experience an increase in the first 120 steps that can cause congestion in two controlled cells (cell 9 and 12). Compared with demand profile 1, more severe congestion that influences two sections (2 and 3) will occur under demand profile 2.

Non-controlled situation

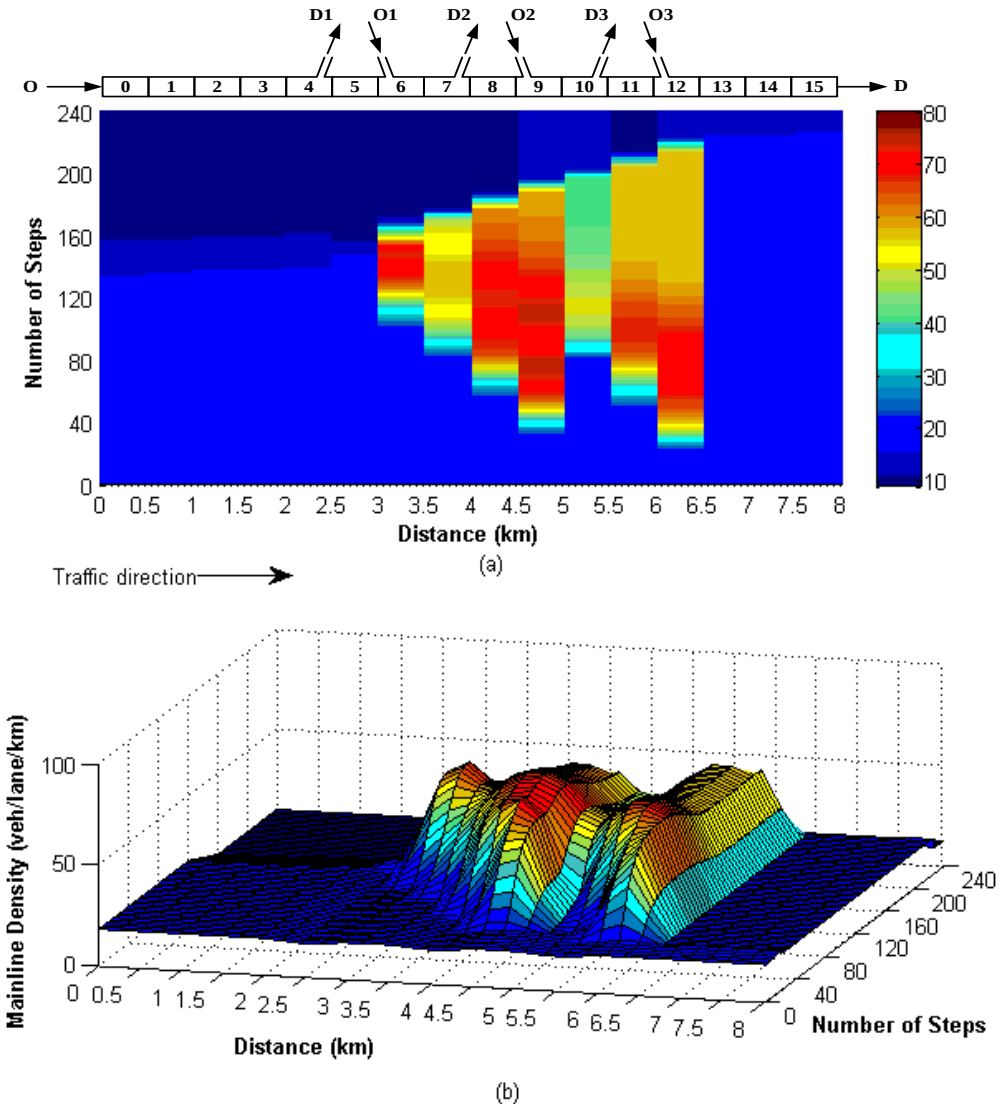


Figure 6.24: Density evolution under NC (demand 2)

Figure 6.24 shows the density evolution in the non-controlled situation. Two congested motorway sections, one starting from cell 9 and the other originating from cell 12, appear in the network. This traffic congestion starts from step 20 and lasts around 200 time steps. After the 220th time step, congestion dissipates completely. Under demand profile 2, the congestion is more severe than the first demand profile from both temporal and spatial points of view. Without control, the network TTS in this scenario is 36378 veh.min, which is much higher than the first demand situation with only one congested motorway section.

Controlled by RAS

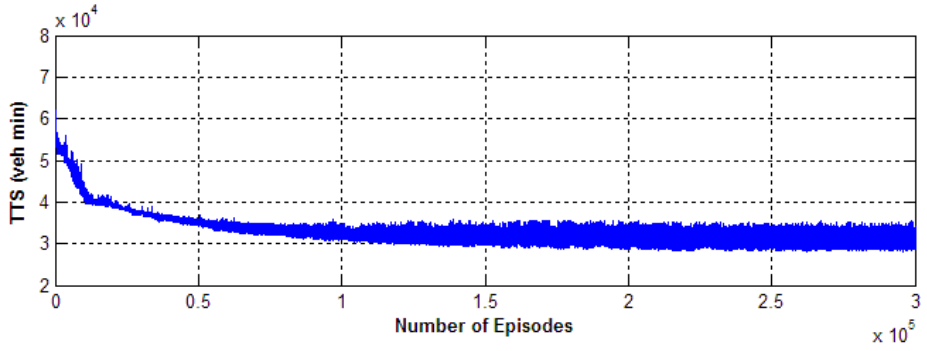


Figure 6.25: RAS convergence (demand 2)

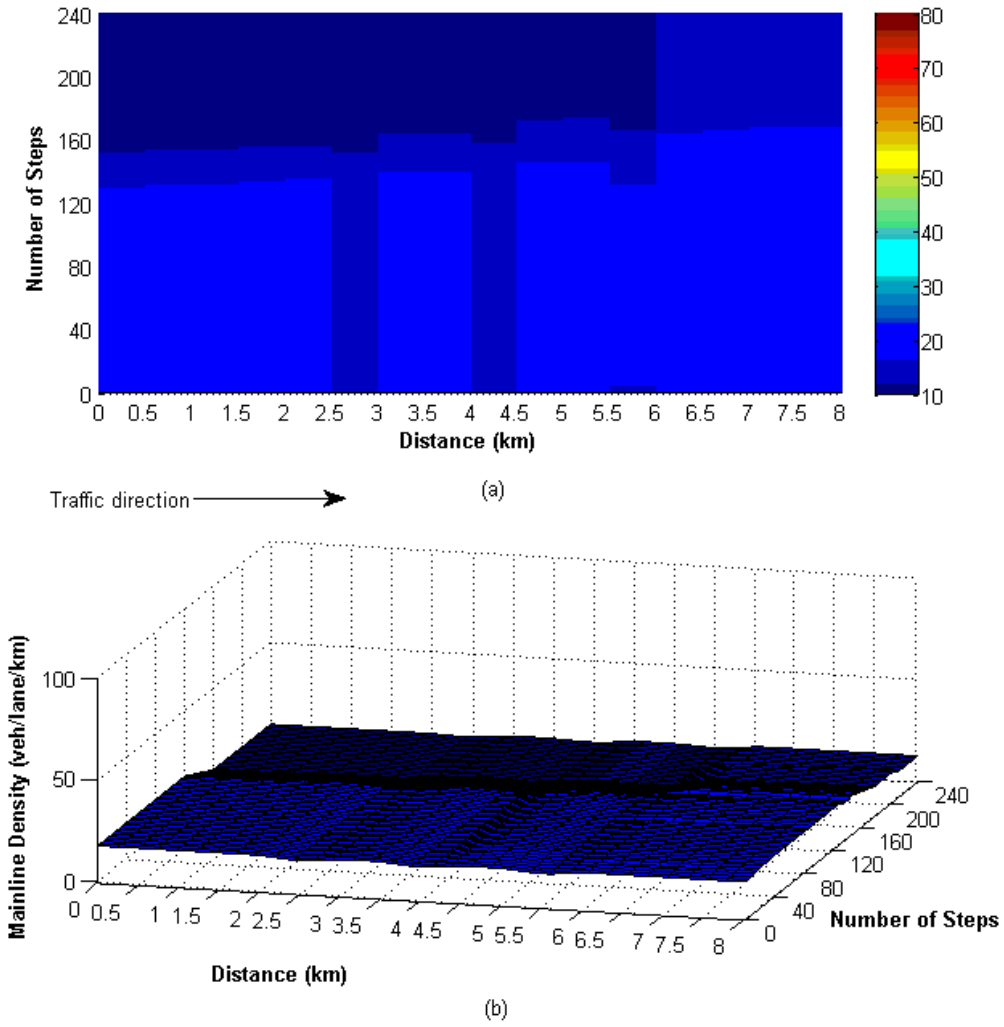


Figure 6.26: Density evolution under RAS (demand 2)

Using the same parameters selected from the first demand profile, RAS also takes about 250000 episodes to find the optimal solution in this test. However, compared with demand profile 1, the convergence of RAS under

demand profile 2 is more unstable (see Figure 6.25). This is mainly because two congested motorway sections are included in this test, and the instability is cumulated from these two sections. Although more severe congestion that affects two motorway sections occurs, RAS can still eliminate this congestion and keep the mainline density around the optimal value as presented in Figure 6.26. Consequently, the network TTS can be reduced to 27482 veh.min.

Comparison with ALINEA

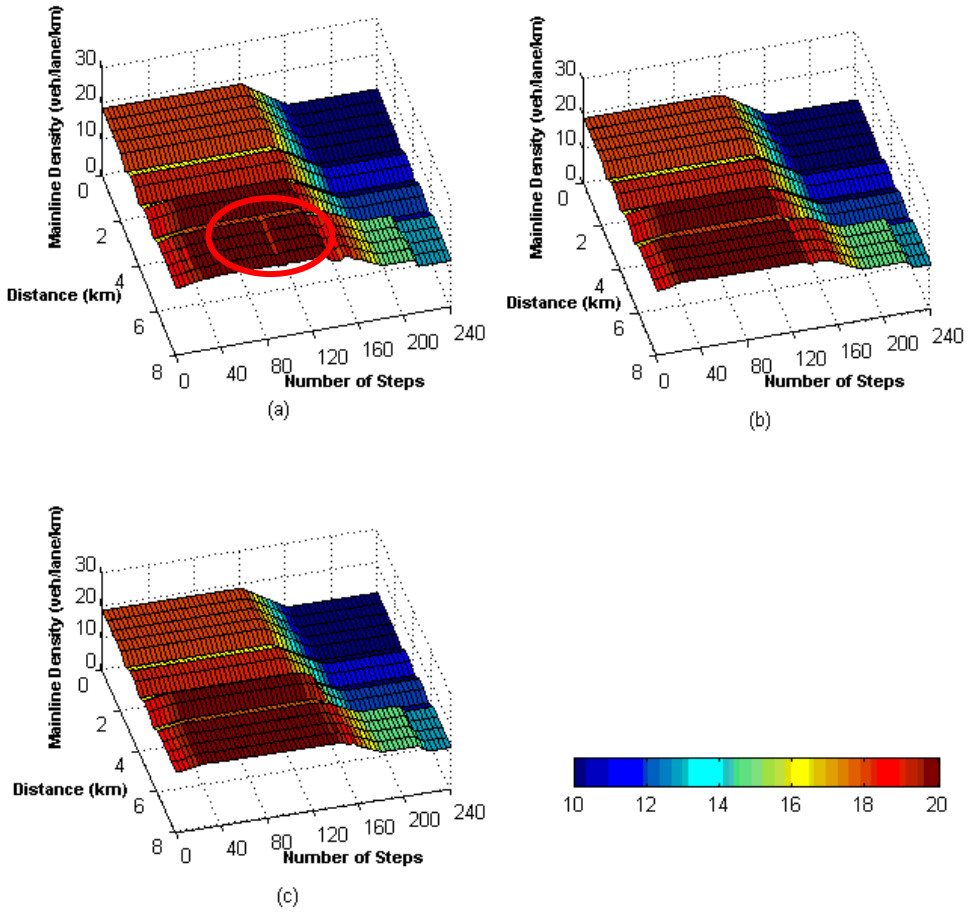


Figure 6.27: Density comparison (demand 2) for: (a) RAS, (b) ALINEA-C, (c) ALINEA-D

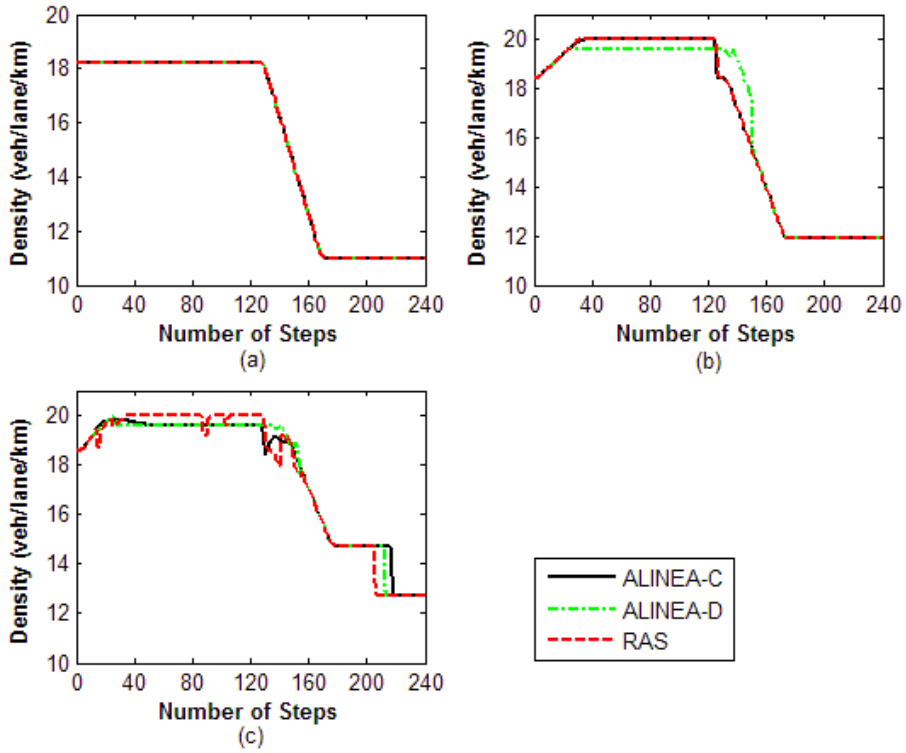


Figure 6.28: Cell density (demand 2): (a) cell 6, (b) cell 9, (c) cell 12

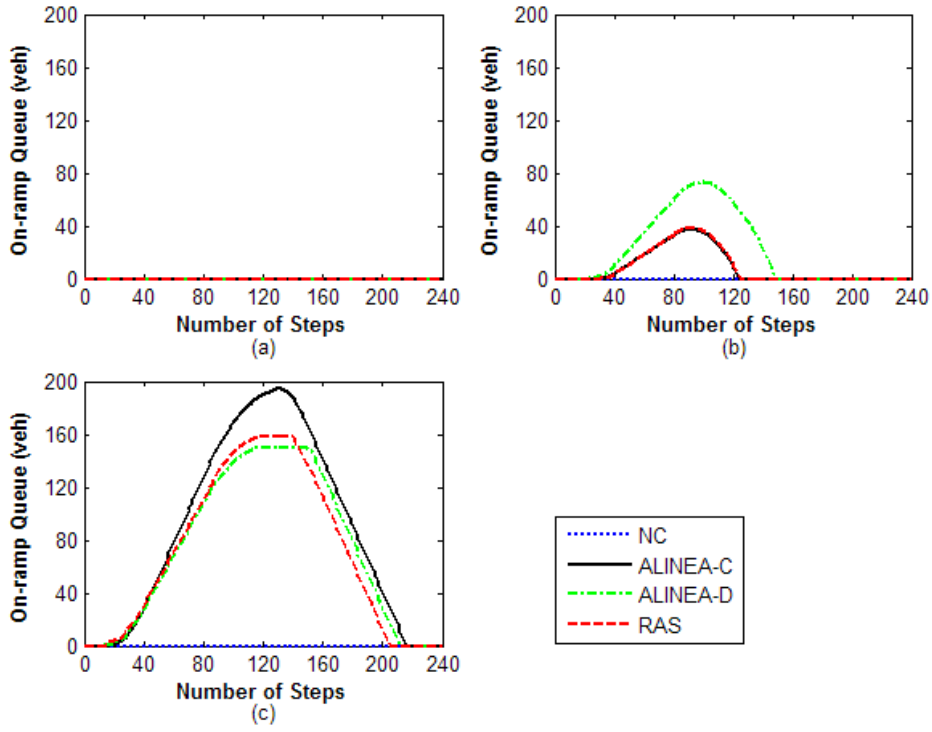


Figure 6.29: On-ramp queue comparison (demand 2) for: (a) on-ramp 1, (b) on-ramp 2, (c) on-ramp 3

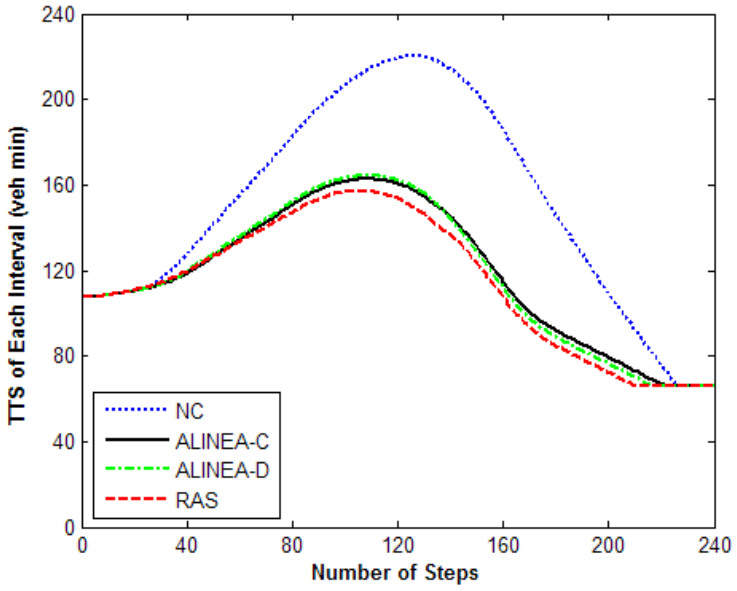


Figure 6.30: Network TTS comparison (demand 2)

Table 6.6: TTS comparison (demand 2)

Strategies	TTS for the whole period (veh.min)						
	Section 0	Section 1	Section 2	Section 3	Section 4	Network	Reduction (%)
NC	5316	4909	10314	11048	4791	36378	–
ALINEA-C	5316	3936	4652	9733	4791	28428	21.9
ALINEA-D	5316	3936	5442	8870	4791	28355	22.1
RAS	5316	3936	4666	8773	4791	27482	24.5

Different from RAS, some parameters selected from the first test are no longer effective for both ALINEA-C and ALINEA-D under demand profile 2. The calibration should be redone for ALINEA to obtain effective parameter settings. The detailed calibration process is presented in Appendix B.2.1, through which the optimal parameters can be determined as: for ALINEA-C, $K_{R,1} = K_{R,2} = K_{R,3} = 0.3$, $\hat{\rho}_1 = \rho_{crit}$, $\hat{\rho}_2 = \rho_{crit}$, $\hat{\rho}_3 = 0.98\,\rho_{crit}$, and for ALINEA-D, $K_{R,1} = K_{R,2} = K_{R,3} = 0.3$, $\hat{\rho}_1 = \rho_{crit}$, $\hat{\rho}_2 = \rho_{crit}$, $\hat{\rho}_3 = 0.97\,\rho_{crit}$.

Compared with ALINEA-C and ALINEA-D, RAS is more unstable on maintaining mainline density in motorway section 3 (distance around 6 km,

see Figures 6.27 (a)). However, as shown in Figure 6.28, it can keep the mainline density closer to the critical value in the controlled cells 9 and 12. Thus, RAS has higher outflows on the mainline with less vehicles waiting at on-ramps 2 and 3 (see Figure 6.29). As on-ramp 1 cannot cause congestion, all three strategies generate the maximum metering rates for this on-ramp and let vehicles from on-ramp 1 enter the mainline freely without queuing.

Although ALINEA-C can keep a short queue length (the same as RAS) at on-ramp 2, it leads to the longest queue length at on-ramp 3 among the three strategies. In contrast, ALINEA-D has a shot queue length at on-ramp 3, while lets the longest queue form at the on-ramp 2. Accordingly, for reducing the network TTS, ALINEA-C and ALINEA-D possess a similar performance with a reduction of 21.9% and 22.1% respectively (as illustrated in Table 6.6 and Figure 6.30). On the other hand, RAS has the lowest network TTS of the three strategies, which corresponds to a 24.5% reduction from the non-controlled situation.

6.2.4 Efficiency test III

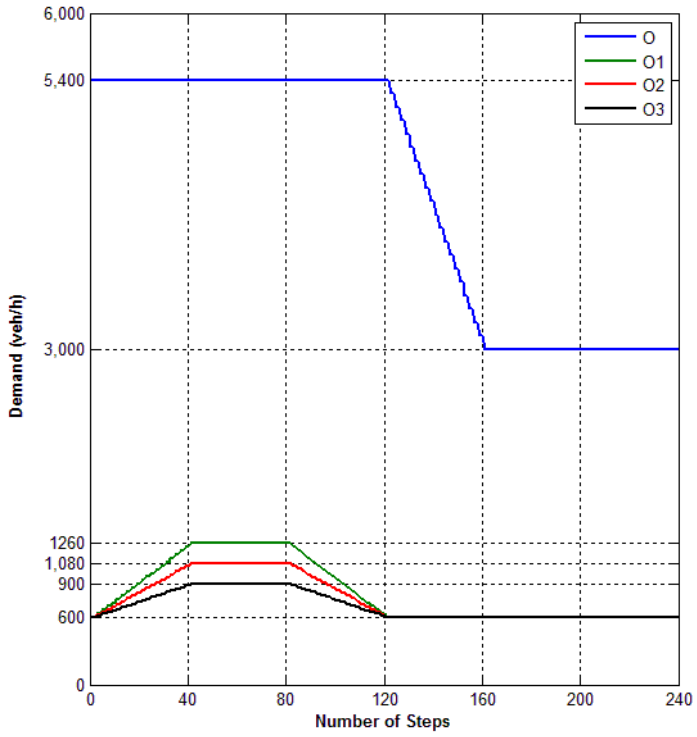


Figure 6.31: Demand profile 3

The mainline demand flow in this test is the same as the previous two scenarios, while compared with demand profile 2, one more on-ramp (on-ramp 1) will have increased demand in the first 120 steps as shown in Figure 6.31. Thus, under demand profile 3, all three on-ramp demand flows are able to cause congestion which will affect three motorway sections (1, 2 and 3). This is the most congested scenario among all three demand profiles.

Non-controlled situation

Figure 6.32 illustrates the density evolution when no control strategies are triggered. Without control, traffic congestion occurs in motorway sections 1, 2 and 3 around the 20th step, which leads to three congested areas during the next 200 steps. Compared with demand profiles 1 and 2, demand profile 3 causes the most serious congestion covering almost 5 km of the 8 km long

motorway. In this situation, the network TTS reaches 40237 veh.min, which is also the highest one among all three demand profiles.

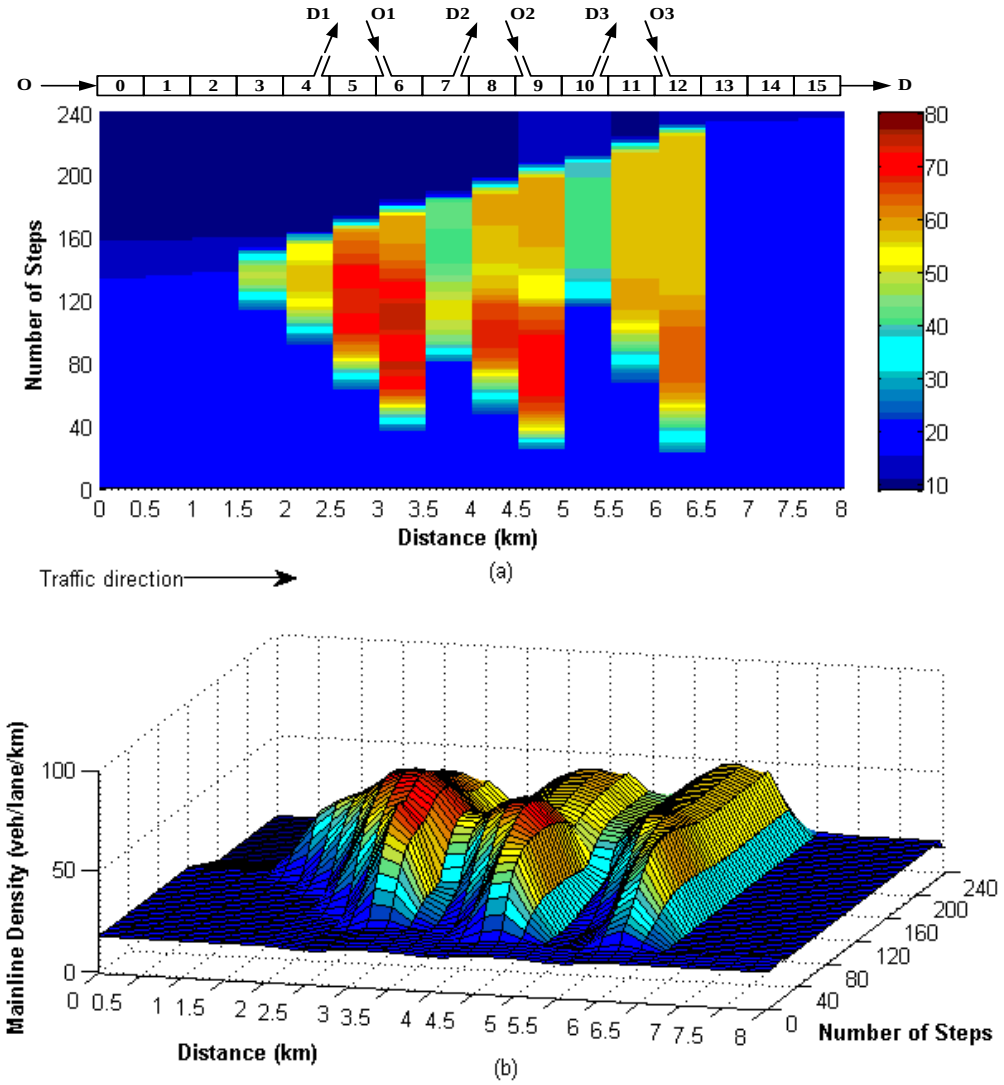


Figure 6.32: Density evolution under NC (demand 3)

Controlled by RAS

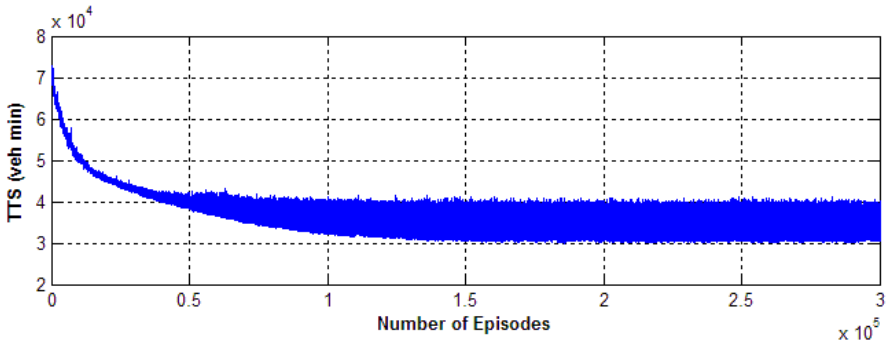


Figure 6.33: RAS convergence (demand 3)

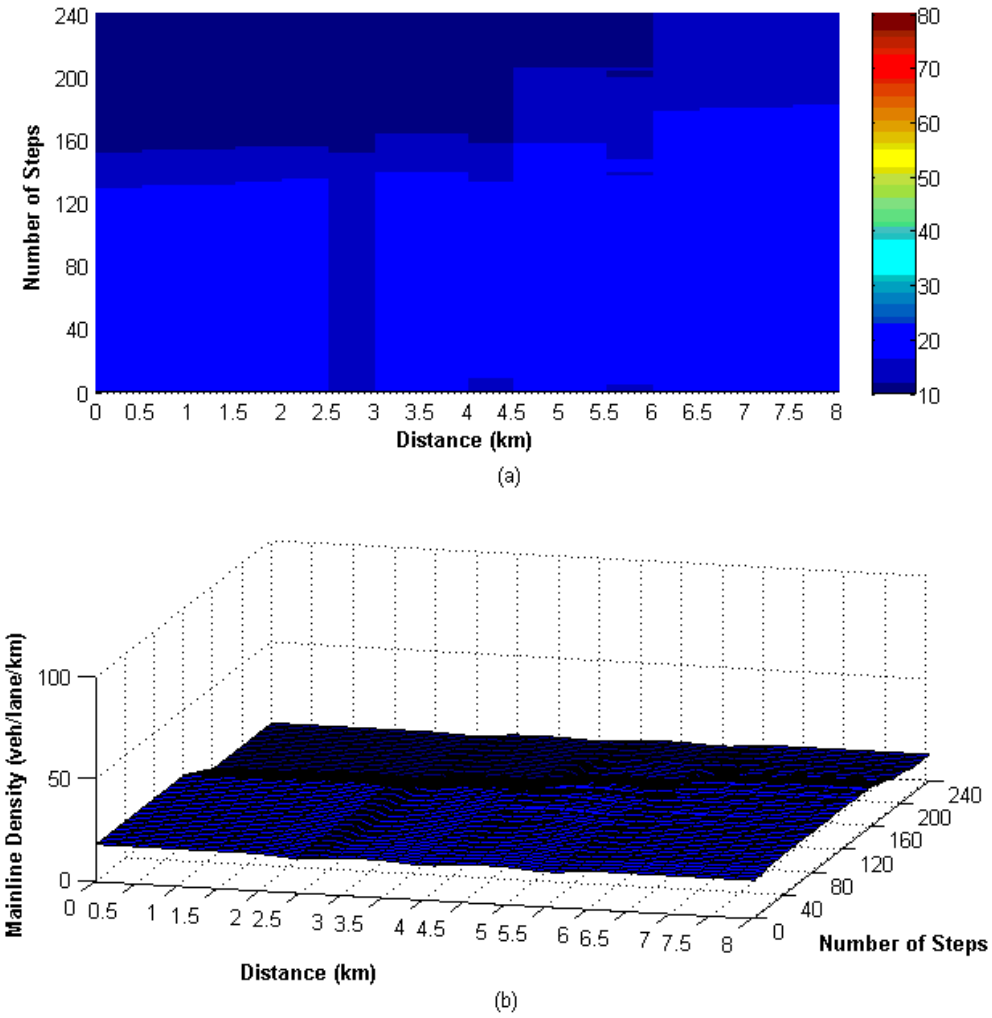


Figure 6.34: Density evolution under RAS (demand 3)

Parameters of RAS are still kept unchanged here. With these parameters, the learning speed of RAS in the third test is almost the same as the previous two scenarios where 250000 episodes are needed to learn the optimal control actions. As shown in Figure 6.33, with three congested motorway sections, demand profile 3 leads to the most unstable convergence among the three scenarios. Although high instability exists, RAS still works well on reducing TTS. Using optimal control actions, RAS can completely eliminate severe congestion (as shown in Figure 6.34) and reduce the network TTS to 30124 veh.min.

Comparison with ALINEA

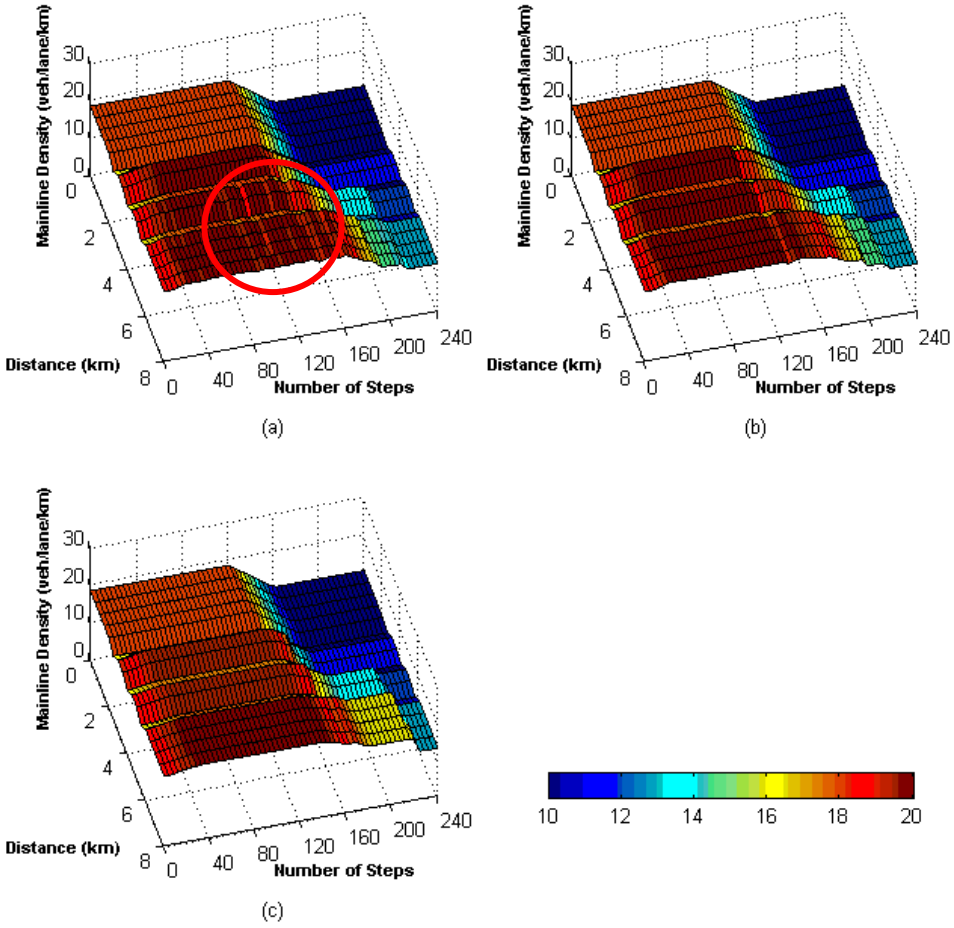


Figure 6.35: Density comparison (demand 3) for: (a) RAS, (b) ALINEA-C, (c) ALINEA-D

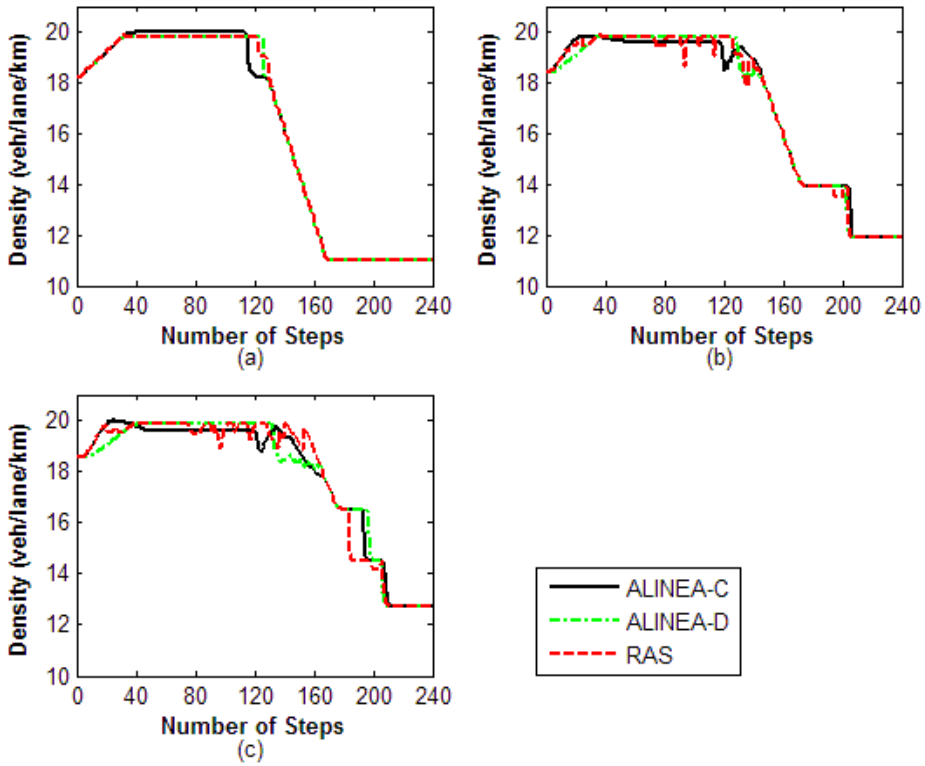


Figure 6.36: Cell density (demand 2): (a) cell 6, (b) cell 9, (c) cell 12

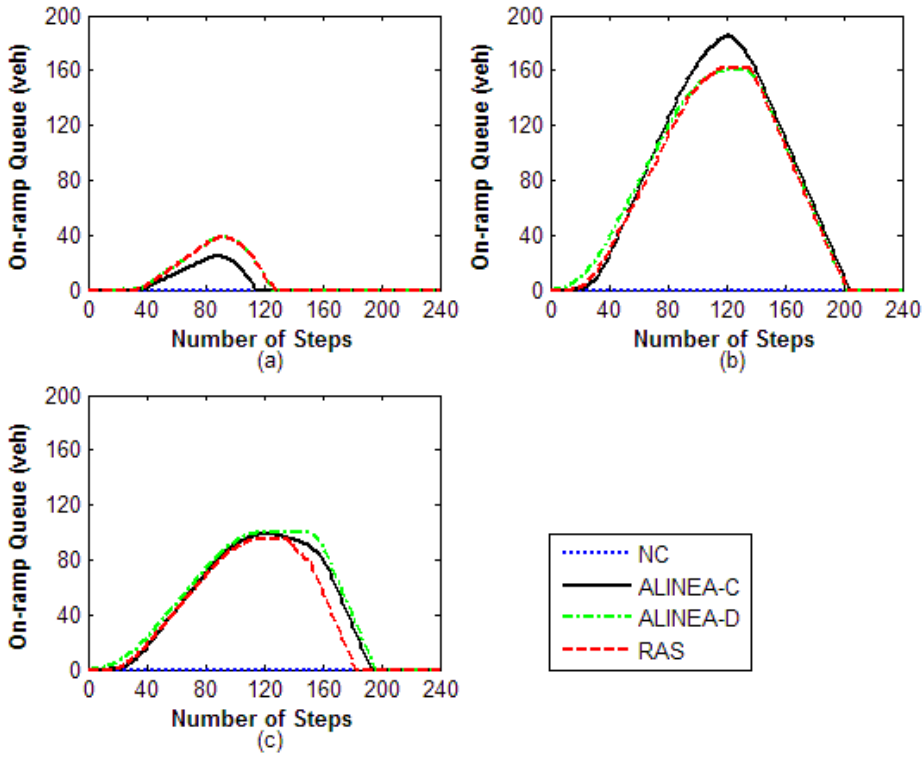


Figure 6.37: On-ramp queue comparison (demand 3) for: (a) on-ramp 1, (b) on-ramp 2, (c) on-ramp 3

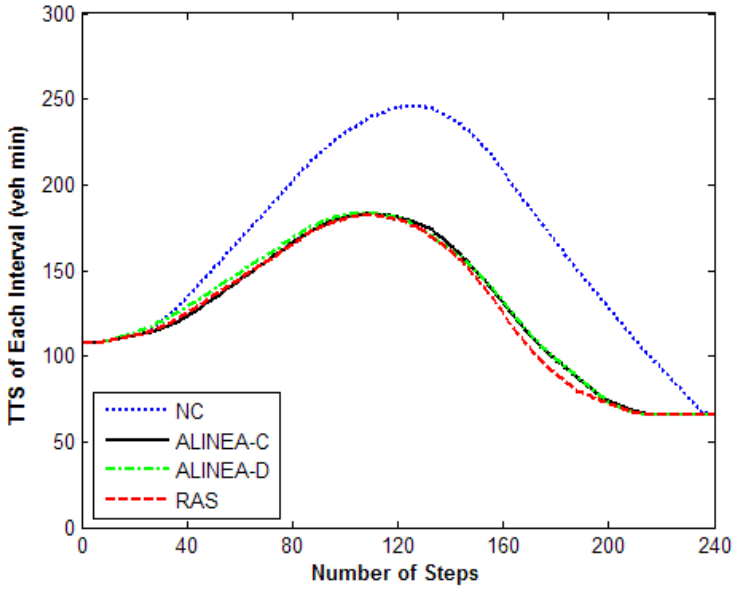


Figure 6.38: Network TTS comparison (demand 3)

Table 6.7: TTS comparison (demand 3)

Strategies	TTS for the whole period (veh.min)						
	Section 0	Section 1	Section 2	Section 3	Section 4	Network	Reduction (%)
NC	5650	9116	10362	10261	4848	40237	–
ALINEA-C	5316	4280	8898	7199	4848	30541	24.1
ALINEA-D	5316	4510	8697	7439	4848	30811	23.4
RAS	5316	4510	8529	6921	4848	30124	25.1

Some parameters of both ALINEA-C and ALINEA-D used in the previous sections are not suitable here, because of the changed demand profile. A recalibration of these parameters is shown in Appendix B.2.2. Then, the optimal parameter values is listed below: for ALINEA-C, $K_{R,1}$ = 0.3, $K_{R,2}$ = 0.3, $K_{R,3}$ = 0.1, $\hat{\rho}_1 = \rho_{crit}$, $\hat{\rho}_2$ = 0.98 ρ_{crit}, $\hat{\rho}_3$ = 0.98 ρ_{crit}, and for ALINEA-D, $K_{R,1}$ = 0.3, $K_{R,2}$ = 0.2, $K_{R,3}$ = 0.1, $\hat{\rho}_1 = \rho_{crit}$, $\hat{\rho}_2$ = 0.96 ρ_{crit}, $\hat{\rho}_3 = \rho_{crit}$.

As shown in Figure 6.35, under demand profile 3, both densities of motorway sections 2 (distance around 4 km) and 3 (distance around 6 km) experience an unstable evolution when RAS is triggered. On the other hand,

ALINEA-C and ALINEA-D can maintain a smoother density evolution in all three controlled motorway sections.

Although RAS cannot make a smooth density evolution, it can keep the densities of cells 9 and 12 closer to the critical value compared with ALINEA-C and ALINEA-D (see Figure 6.36). Therefore, similar to the second test, RAS has the shortest queue length at both on-ramps 2 and 3 (see Figure 6.37). Under such circumstances, RAS achieves the lowest network TTS as shown in Figure 6.38, which corresponds to a 25.1% decrease from the non-controlled situation (illustrated in Table 6.7). ALINEA-C and ALINEA-D perform worse with a reduction of 24.1 % and 23.4% respectively.

6.2.5 Equity test

So far, the ability of RAS in improving motorway efficiency (by reducing TTS) has been tested in three scenarios with different demand profiles. In this section, the equity issue will be discussed. As shown in efficiency test 1 (Section 6.2.2), to maintain a high efficiency, vehicle queues only form at the on-ramp 3, and users from on-ramps 1 and 2 can access the motorway mainline freely without any restriction. The unfairness of using the motorway mainline in this case is very obvious, as only the users from on-ramp 3 have to wait at the on-ramp, and users from the other two on-ramps have no waiting times at all. This demand profile will be used to test the ability of RAS on maintaining user equity.

Controlled by RAS-EQ

In this test, equity is added as an additional objective and two control objectives regarding traffic efficiency and user equity are considered by RAS simultaneously. The multi-objective algorithm developed in Section 4.4.2 is used here to balance efficiency and equity. As only one more objective

(equity) is included, the multi-objective algorithm used here is named RAS-EQ for ease of comparison.

In this test, all three ramp agents have the same weight values for the efficiency and equity. Assume that δ_1 is the weight value of the efficiency-related objective and δ_2 is the weight value of the equity-related objective, then $\delta_1, \delta_2 \in [0,1]$ and $\delta_1 = 1 - \delta_2$. By regulating δ_2 within its permitted range, different importance can be assigned to user equity. In this test, the equity is measured by the standard deviation of total waiting time ($SD(TWT)$ and $SD^t(TWT)$) on different on-ramps, which can be obtained from Equations (4.21) and (4.22) (introduced in Section 4.3.1).

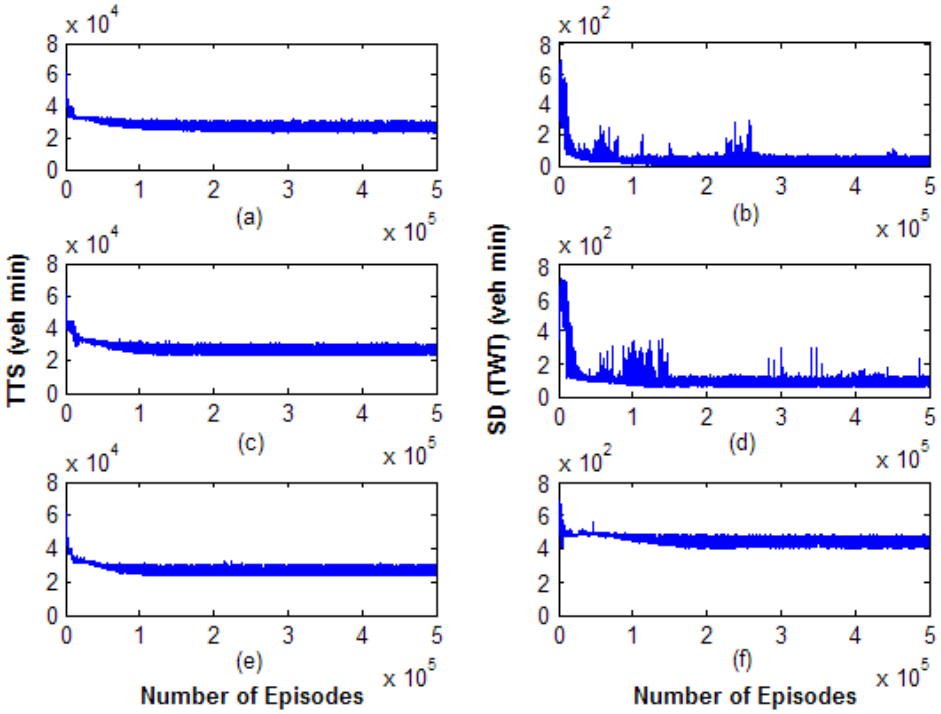


Figure 6.39: RAS-EQ convergence with different δ_2: (a) 0.9, (b) 0.9, (c) 0.5, (d) 0.5, (e) 0.1, (f) 0.1

Table 6.8: TTS and SD (TWT) under different δ_2

δ_2	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
TTS (veh.min)	23195	24038	24047	24059	24156	24187	24213	24385	24452	24566	28809
Minimum SD(TWT) (veh.min)	424.2	412.3	392.7	306.1	88.5	57.5	14	2.7	1	0.2	0

Table 6.8 illustrates the network TTS and on-ramp SD(TWT) under different values of δ_2. Since this test is focused on the equity, the minimum SD(TWT) after convergence (300000 episodes as shown in Figure 6.39) that corresponds to the highest equity and its related TTS are selected.

For reward normalisation, the value of $SD_{ef}(TWT)$ in Equation (4.33) (introduced in Section 4.3.3) can be determined by the maximum $SD^t(TWT)$ under the control of efficiency-orientated RAS, which is 6.7 veh.min in this case (see Figure 6.41).

When δ_2 = 1, equity will be the only objective considered. This is the same as the non-controlled situation, in which users can enter motorway mainline freely and thus have no waiting times at all on-ramps. Thus, SD(TWT) in this case is 0. When δ_2 = 0, equity will not be considered and RAS becomes an efficiency-oriented algorithm as discussed in the previous section. Higher δ_2 corresponds to higher TTS and smaller SD(TWT), as more importance is assigned to user equity. This result is consistent with the findings mentioned in (Kotsialos and Papageorgiou 2004a, Meng and Khoo 2010, Zhang and Levinson 2005) where user equity was found to be partially competitive with traffic efficiency, and the most efficient case is also the most inequitable case (as illustrated in Table 6.8).

Comparison with other strategies

RAS-EQ with a high equity (δ_2=0.9) is used to make a comparison with efficiency-oriented RAS (i.e. RAS) and ALINEA. Figure 6.40 shows on-ramp queues under different control strategies. Compared with RAS and ALINEA which only control vehicles from on-ramp 3, RAS-EQ makes a more even distribution of queues at different on-ramps. Thus, during the test period, especially the first 120 steps with heavier traffic load, RAS-EQ can maintain a lower SD(TWT), below 2 veh.min at each time step, than other strategies (see Figure 6.41). For SD(TWT) of the whole test period, RAS-EQ has a very low value, 0.2 veh.min, which is close to the no controlled situation and much lower than ALINEA-C (221.8 veh.min), ALINEA-D (434.1 veh.min) and RAS (424.2 veh.min).

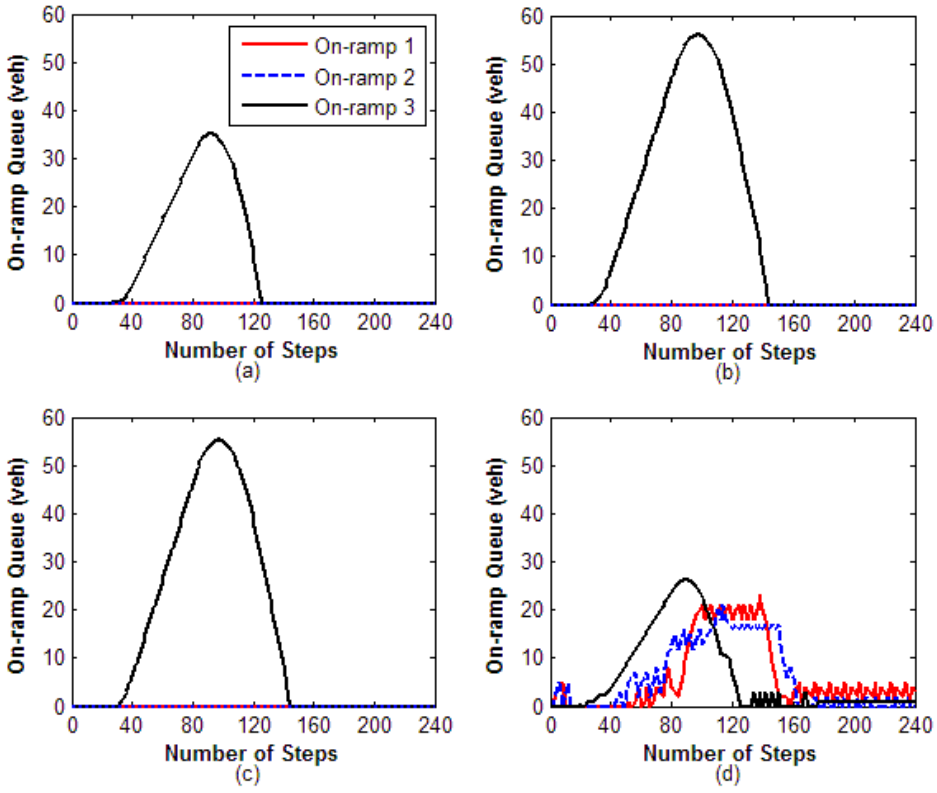


Figure 6.40: On-ramp queues comparison: (a) ALINEA-C, (b) ALINEA-D, (c) RAS, (d) RAS-EQ

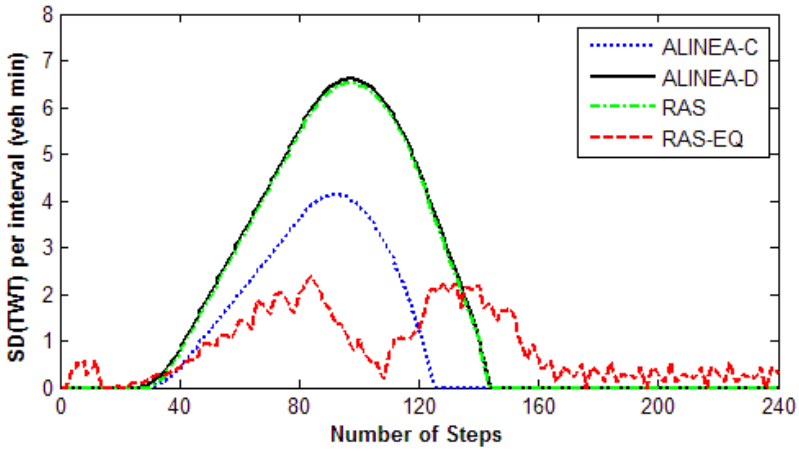


Figure 6.41: SD(TWT) comparison

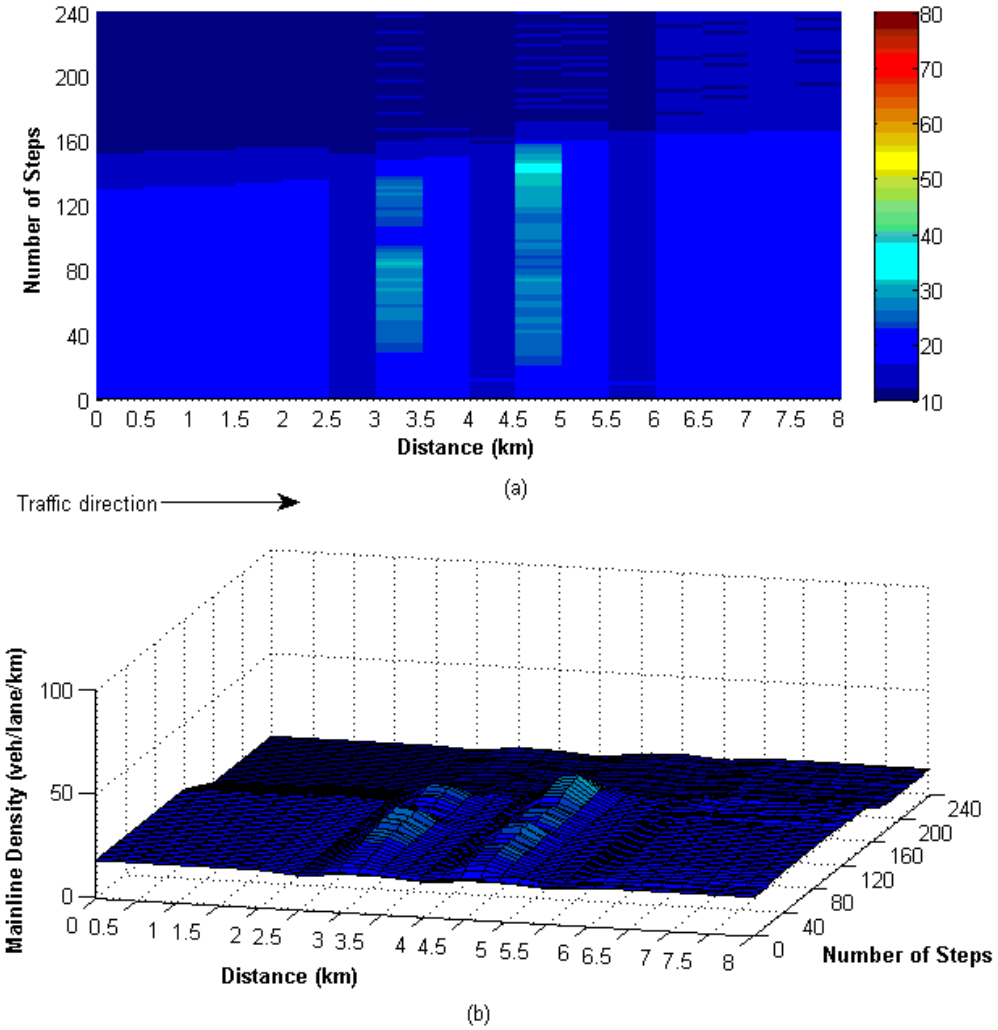


Figure 6.42: Density evolution under RAS-EQ

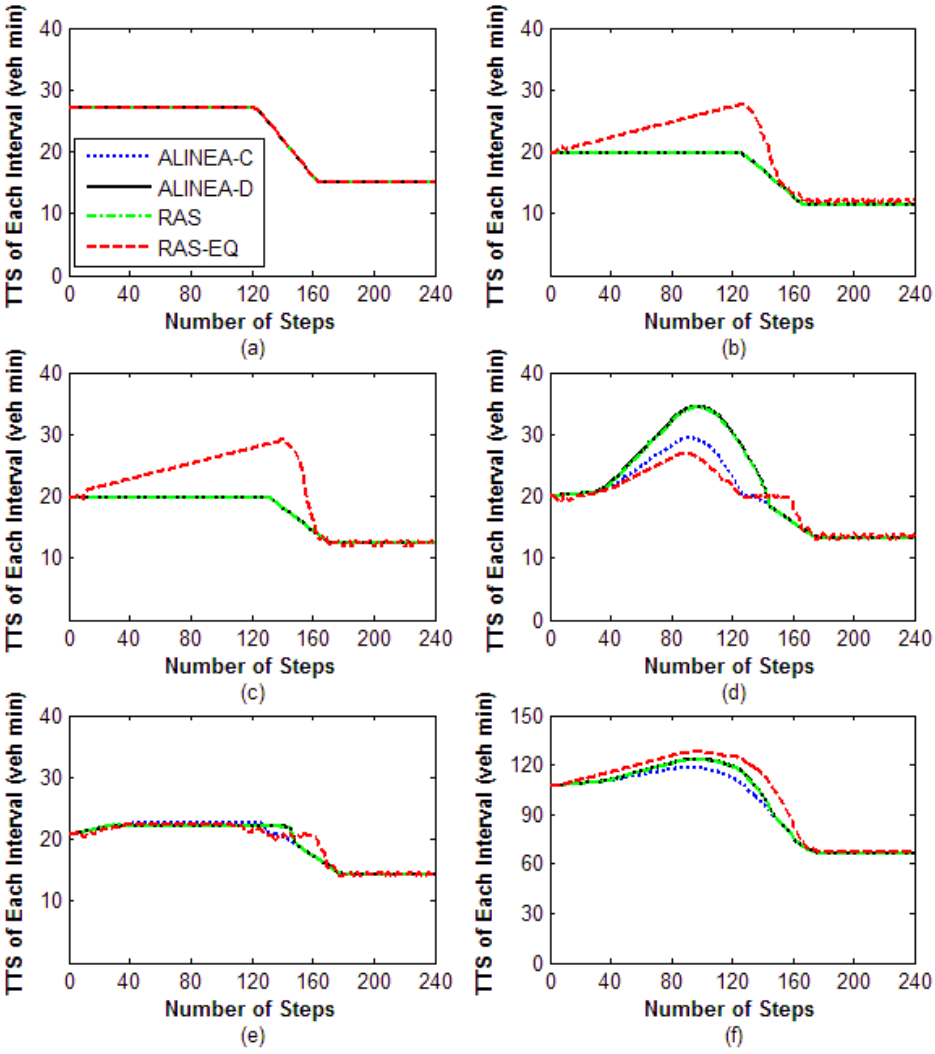


Figure 6.43: TTS comparison for equity test: (a) section 0, (b) section 1, (c) section 2, (d) section 3, (e) section 4, (f) network

Table 6.9: TTS and SD(TWT) under different strategies

Strategies	NC	ALINEA-C	ALINEA-D	RAS	RAS-EQ
TTS (veh.min)	28809	22752	23202	23195	24566
TTS reduction (%)	–	21.0	19.5	19.5	14.7
SD(TWT) (veh.min)	0	221.8	434.1	424.2	0.2

Because of the fairly distributed TWT, the TTS of the three motorway sections (1, 2 and 3) controlled by RAS-EQ is also equally distributed. On the other hand, as shown in Figure 6.43 (b), (c) and (d), RAS and ALINEA make the TTS of motorway section 3 much higher than the other two

controlled sections. To maintain a higher equity, RAS-EQ at on-ramps 1 and 2 leads to slight congestion in cell 6 (between 3 and 3.5 km) and 9 (between 4.5 and 5 km) at some time steps by managing equal on-ramp queues (Figure 6.40). Thus, compared with RAS and ALINEA, RAS-EQ has the highest TTS of the whole network (Figure 6.43 (f)).

Although RAS-EQ cannot work as well as other strategies in improving traffic efficiency, it can still reduce TTS by 14.7% compared with the non-controlled situation, and in the meanwhile, it can maintain a high equity with very low SD(TWT) (as illustrated in Table 6.9).

6.2.6 Summary

In this section, the performance of RAS consisting of multiple ramp agents was tested in an extended network with multiple on-ramps. Three demand profiles with different congestion levels were used in the test.

Under demand profile 1, only one on-ramp may cause congestion. A similar result as the single-ramp case was found in this test. With discrete metering rates, RAS achieved almost the same performance of ALINEA-D which reduced the network TTS by 19.5% from the no controlled situation. Using continuous metering rates, ALINEA-C showed a better performance with a 21% reduction on the network TTS.

Under demand profiles 2 and 3, more than one on-ramp may cause congestion. RAS in these two scenarios outperformed both ALINEA-C and ALINEA-D on reducing TTS with a reduction of 24.5% and 25.1% respectively. However, RAS could not keep a smooth density evolution as both ALINEA-C and ALINEA-D did.

In the equity test, the demand profile 1 was used for analysis. It can be found from the comparative result that RAS-EQ achieved a superior performance on maintaining user equity than RAS (424.2 veh.min) and

ALINEA (221.8 veh.min for ALINEA-C, 434.1 veh.min for ALINEA-D), which corresponded to a low SD (TWT) of only 0.2 veh.min. In the meanwhile, RAS-EQ could reduce the network TTS by 14.7% from the non-controlled situation. Thus, with suitable weight values, efficiency and equity can be well balanced by RAS-EQ.

6.3 Discussion

In this chapter, two case studies based on hypothetical networks, namely the single-ramp and multi-ramp cases were designed to test the performance of RAS. Some findings obtained from various comparative experiments in terms of the parameter characteristics, algorithm advantages and disadvantages are discussed here:

(1) Learning parameters (α , γ , ε) selected from the single-ramp case worked well in all tests presented in this chapter. Indeed, it has been found in Section 6.1.3 that, within the test range, these three parameters had no obvious effects on the optimal solution itself (the benchmark line), and they only affected how fast and steadily this solution can be obtained. Thus, without recalibrations, RAS can be directly used in both the single-ramp case and multi-ramp case, which means RAS has good adaptability in a new environment (as introduced in Chapter 1 and mentioned in some other studies such as (Davarynejad et al. 2011, Jacob and Abdulhai 2010, Rezaee et al. 2012)). On the other hand, the parameters of ALINEA need to be recalibrated to obtain an acceptable performance in some scenarios, such as the multi-ramp case with demand profile 2 and 3.

(2) In some cases with light congestion such as the single-ramp case and the multi-ramp case under demand 1, RAS achieved almost the same performance as ALINEA-D in reducing TTS. With continuous metering

rates in this situation, ALINEA-C was even a little better than RAS. When heavier congestion that may influence more than one on-ramp occurs, RAS obtained a superior performance than either ALINEA-C or ALINEA-D. This means, compared with ALINEA, RAS has no improvement on reducing TTS when one ramp agent can work independently to eliminate congestion. On the other hand, when multiple ramp agents should work together to alleviate congestion, RAS outperforms ALINEA.

(3) When the equity issue was considered, RAS-EQ could successfully maintain a low SD (TWT) close to the non-controlled situation, which was much better than ALINEA. With different weight values, RAS could balance efficiency and equity to different degrees. In this test, it was found that higher efficiency usually corresponded to lower equity, which reaffirms the competitive relationship of efficiency and equity found in previous studies (Kotsialos and Papageorgiou 2004a, Meng and Khoo 2010, Zhang and Levinson 2005).

(4) Although many benefits regarding improving efficiency and maintaining equity can be achieved by RAS, one drawback was found in the tests carried out in this chapter. As presented in Sections 6.2.3 and 6.2.4, when severe congestion occurs, RAS failed to keep a smooth density evolution as ALINEA did. This is mainly because RAS adopted a uniformly divided discrete state space. In the severe congested situation, mainline density will always fluctuate around the critical value, and slight changes of mainline density around this value may lead to completely different TTS. In the uniformly divided state space, states around the critical value may not be enough to distinguish the slight changes of mainline density, and thus make the control strategy unstable. The other reason for this is the action selection strategy RAS used. As mentioned in Section 4.3.4, there exists a probability for the ramp agent to select

new actions that may be non-greedy. Sometimes a worse attempt may make the control system unstable at some points. However, it was found from various tests that the unstable density evolution had no obvious effects on the network TTS. Under this circumstance, if a reduction in TTS is the main concern, this kind of instability can be ignored. However, if the system stability is considered as one important issue, other kinds of state space or adaptive action selection strategies can be analysed in future work.

CHAPTER 7 CASE STUDY FOR REAL NETWORK

Chapter 6 has shown the test of RAS in two hypothetical cases with simplified demand profiles. In this chapter, the third case adopting a real network is analysed. This case is named the real-network case which focuses on a practical motorway network with real observed traffic data. The realistic network layout and fluctuating traffic flows are considered in this case, which will be used to test RAS regarding its ability to: (1) improve traffic efficiency and (2) maintain user equity with a highly fluctuating demand profile and split ratios.

Chapter 7 is organised as follows. Section 7.1 describes the real network selected for the test. Then, how ACTM is calibrated to simulate the traffic in this network is discussed in Section 7.2. The validation of calibrated ACTM is presented in Section 7.3, which shows the effectiveness of the calibration. After that, RAS is tested using ACTM calibrated from the real traffic data in Section 7.4. Section 7.5 finally summarises this chapter.

7.1 Description of the Real Network

7.1.1 Network layout

One of the metered motorway segments (southbound direction) of the M6 in the UK was chosen for the case study. This motorway segment is between junction 10A (J10A) and junction 6 (J6) with an approximate length of 11.6 km (outlined in blue in Figure 7.1). The reasons for selecting this network are summarised as follows:

(1) A part of this network has been analysed in (Bergsma 2006) where recurrent congestion was found during peak hours. This can provide an effective scenario to test the proposed system.

(2) High quality flow and speed data are available, which can be used to set parameters for ACTM and test RAS.

(3) Various layouts of on-ramps are included in the network, such as metered on-ramps in J10, J9 and J7, unmetered on-ramp in J8, closely located on-ramps in J10 and J9, and an isolated on-ramp located at J7, which provides a more general case to test RAS.

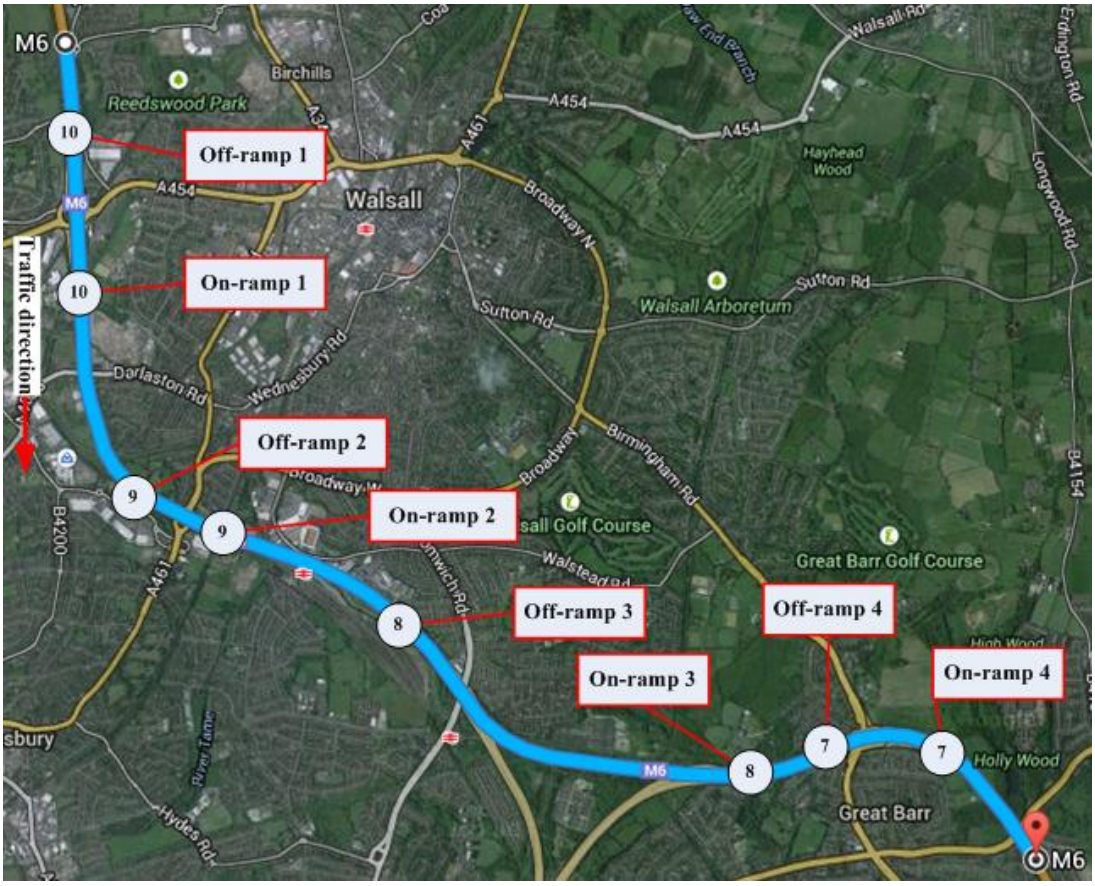


Figure 7.1: Real network layout

7.1.2 Network partition

The test motorway network consists of a three-lane mainline, three metered two-lane on-ramps, one unmetered on-ramp and four off-ramps. According

to the location of ramps and road layout, the whole motorway segment is divided into 21 cells with indices from 0 to 20. Among these cells, cells 5, 9 and 18 are controlled cells linked with three metered on-ramps, i.e. on-ramp 1, 2 and 4. The network partition and length of each cell can be seen from Figure 7.2.

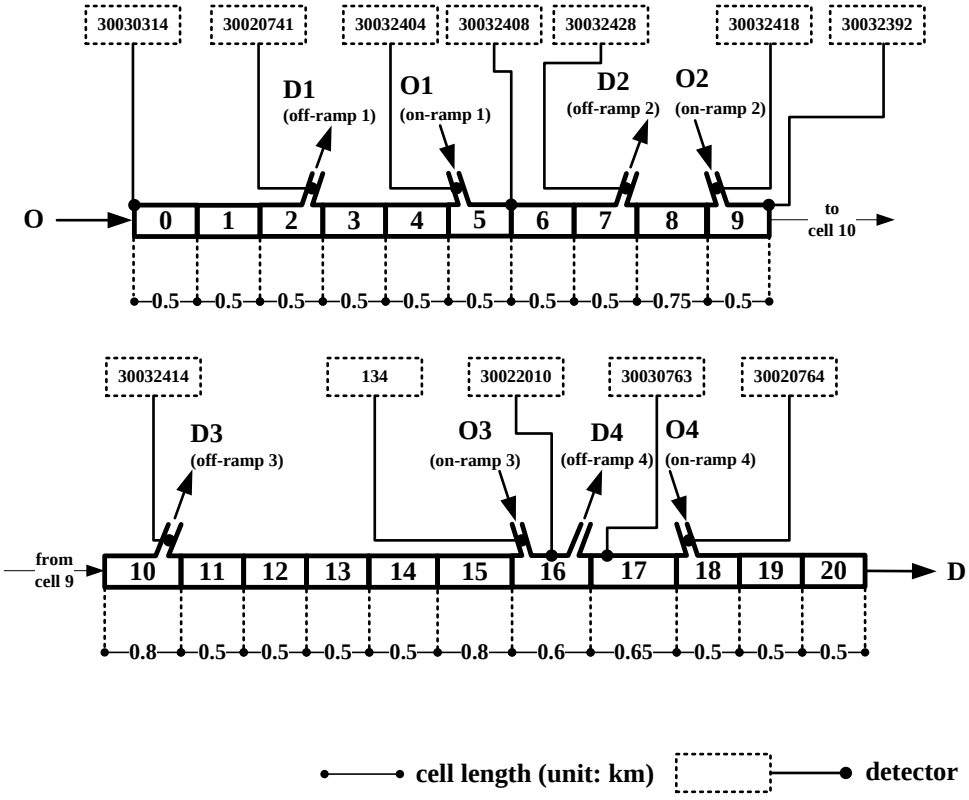


Figure 7.2: Network partition

7.2 Parameter Settings for ACTM

7.2.1 Available data

Two kinds of data recorded in the highways agency database JTDB (journey time database) and TRADS (traffic flow data system) [7] can be obtained to set parameters for ACTM and test RAS. JTDB provides link data between

[7] These two databases are included in the highways agency traffic information system (HATRIS), which can be found from: https://www.hatris.co.uk/ .

two adjacent junctions, which can be used to calibrate the fundamental diagram and generate suitable values for the parameters of ACTM. TRADS contains detector data from the selected motorway network. Real demand flows and split ratios at different times can be generated from these data, which can be used to set the simulation experiments and test the effectiveness of selected parameters. Examples of the data collected from JTDB and TRADS can be found in Appendix C.

As introduced in Section 4.1, ACTM is derived from the fundamental diagram (triangular or trapezoid). Thus, the parameters of ACTM can be obtained from one related fundamental diagram, or specifically the triangular fundamental diagram in this study. Two kinds of link data namely link flow (traffic flow passing through a link between two junctions) and link average speed can be collected from JTDB (Appendix C.1). However, not all time periods have the full data record for the network studied. A time period between May and August 2011 was selected, because it contains all the traffic data required. Furthermore, more than 95% data of this period are marked as "high quality" [8]. Only high quality data are considered in this work.

Density can be calculated by the basic relationship $\rho = q / v$ (introduced in Section 2.1) from flow and speed. Consequently, the scatter plot of flow and density can be obtained (see Figure 7.3). The link data collected between J9 and J8 is used, since this link experiences more severe congestion than others, which can provide better congestion-related flow data.

---

8 The definition of "high quality" can be found from the JTDB reference manual (https://jtdb.hatris.co.uk/JTDB%20Reference%20manual.pdf). Each data source has its own definition of high quality. Take the MIDAS for example (which is the main data source of JTDB), the high quality data should be: (1) observed data, and (2) with a minimum of a loop per 1 km of link.

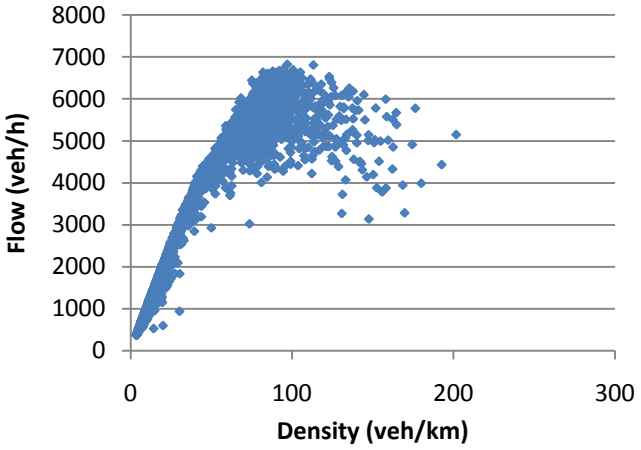


Figure 7.3: Flow-density scatter plot

To form the triangular fundamental diagram, three main parameters: free-flow speed v, congestion wave speed w, and capacity q_{cap} should be determined. Other parameters such as critical density ρ_{cap} and jam density ρ_{jam} can be obtained from the triangular fundamental diagram shown in Figure 7.4.

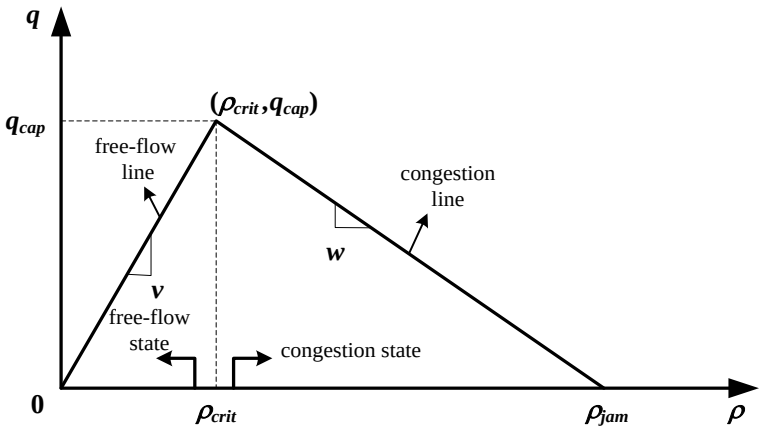


Figure 7.4: Target fundamental diagram

7.2.2 Parameter Settings

Free-flow speed

There are a total of 8928 lines of data among which 8723 lines have the quality “high”, and only high quality data are used. The method mentioned in (Chow and Li 2014) can be used to determine free-flow speed. In their work,

the free-flow line of the fundamental diagram is formed by linear regression, and then the slope of this free-flow line is regarded as free-flow speed (as shown in Figure 7.4). To fit the regression, free-flow related data should be selected from all observed data. Here a threshold of speed is defined as the 85th percentile of all observed speeds, i.e. 98 km/h in this case. All data points corresponding to speeds higher than 98 km/h are considered as free-flow related data (Chow and Li 2014). To guarantee that when the density is zero, flow is also zero, the regression line should pass through the origin point (0,0) as shown in Figure 7.5. The slope of this regression line is determined as the free-flow speed v, which is 100 km/h.

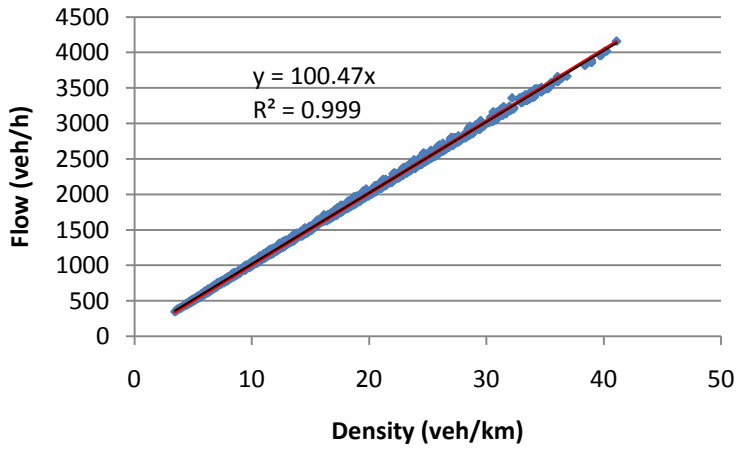


Figure 7.5: Regression of free-flow line

Capacity and critical density

Following (Chow and Li 2014, Dervisoglu et al. 2009), the maximum observed flow 6800 veh/h is regarded as capacity in this study. Given free-flow speed v=100 km/h and capacity q_{cap} = 6800 veh/h, the critical density can be obtained by $\rho_{crit} = q_{cap} / v$ = 68 veh/km.

Congestion wave speed and jam density

Congestion wave speed w is determined from the congestion line of the fundamental diagram (Figure 7.4). The congestion-related data with density higher than the critical density are used to form the congestion line. To guarantee the triangular shape of the fundamental diagram, the constrained regression line should pass the vertex (ρ_{crit}, q_{cap}). From the triangular fundamental diagram shown in Figure 7.4, a relationship between flow and density can be obtained: $(q_{cap} - q) = w \cdot (\rho - \rho_{crit})$. Letting $y = q_{cap} - q$ and $x = \rho - \rho_{crit}$, the constraint regression can be converted to the method used for determining free-flow speed where the regression line should pass through the origin point (0,0).

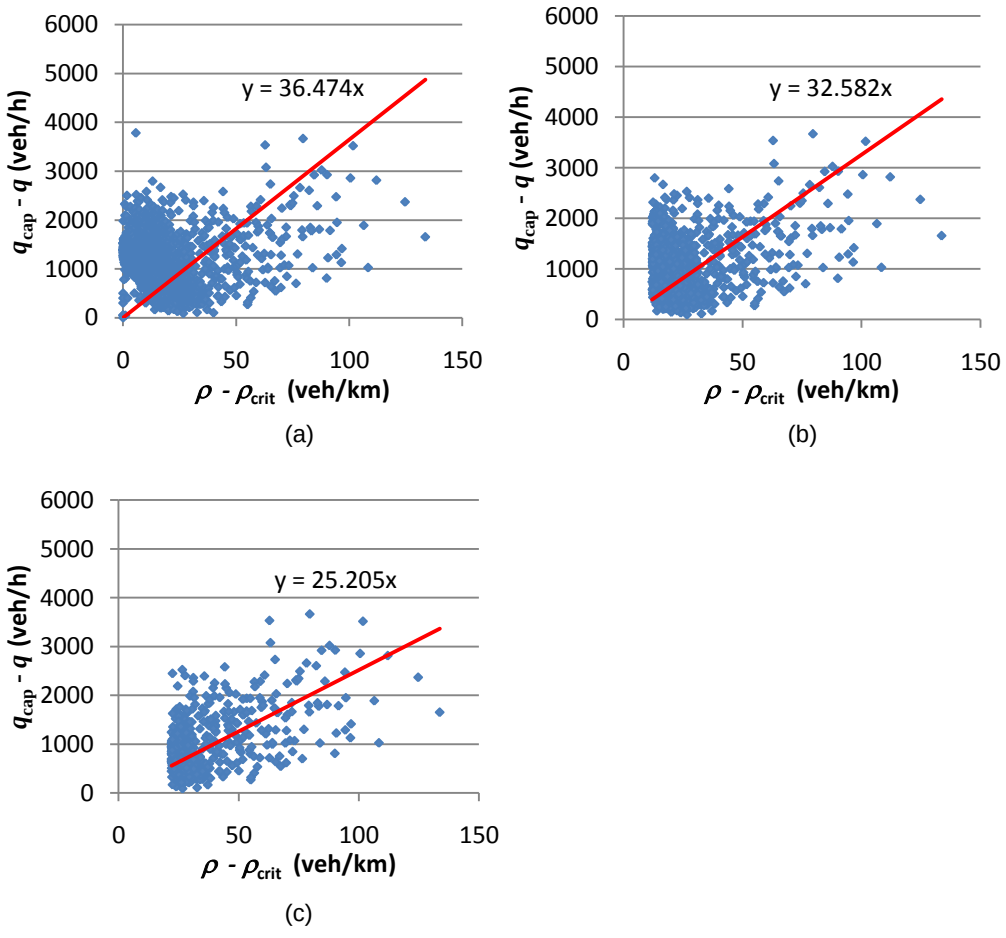


Figure 7.6: Regression of congestion line using data with density larger than: (a) 68 veh/km, (b) 80 veh/km, (c) 90 veh/km

However, not all data that have density larger than the critical value (68 veh/km) can be used to calibrate the congestion line in our case. As shown in Figure 7.6, the data with density around 68 veh/km (between 50 and 90 veh/km) are highly unstable and many data within this range have flows much lower than the capacity (6800 veh/h). Including these data, the regression line may be unrealistic with higher congestion wave speeds (see Figure 7.6 (a)). As mentioned in (Muñoz et al. 2004), a physically reasonable congestion wave speed in the real network should be $10 \le w \le 20$ mile/h ($16 \le w \le 32$ km/h). Thus, only data with density higher than 90 veh/km are used to calibrate the congestion line, and the congestion wave speed (w) can be obtained from Figure 7.6 (c), which is 25 km/h. Jam density can then be calculated by $\rho_{jam} = q_{cap} / w + \rho_{crit}$ = 340 veh/km. Although the data with density higher than 90 veh/km can provide a reasonable congestion wave speed, the regression of these data is not very good with R^2 = 0.25 (shown in Figure 7.6 (c)). It is mainly because the available database JTDB cannot provide enough congestion data (with high density), and most data are distributed around the critical density. However, with 25 km/h as the congestion wave speed, ACTM works well when the averaged midweek data are used to test ACTM in Section 6.3.3. These averaged midweek data will also be used to test RAS. Therefore, the congestion wave speed and resultant jam density obtained in this section can be used to test RAS in Section 6.3.4.

Capacity drop

λ is another important parameter of ACTM which is used to simulate capacity drop phenomenon at bottleneck locations. To obtain the value of this parameter, the flow-density data at bottleneck locations are required. However, only link flow-density data are available so far and they are not

sufficient to determine capacity drop. Under such circumstances, the result presented in a published report (Bergsma 2006) is considered. In this report, a 10% capacity drop was found in the same motorway location. According to this finding, λ can be determined as 0.9 in this study.

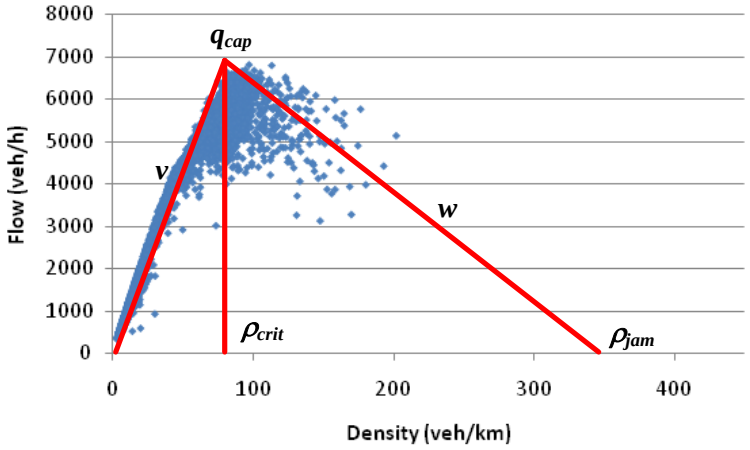


Figure 7.7: Calibrated fundamental diagram

After all the parameters have been decided, the final triangular fundamental diagram can be seen in Figure 7.7. Parameters of ACTM are determined as follows: free-flow speed v is 100 km/h, capacity q_{cap} is 6800 veh/h, congestion wave speed w is 25km/h, critical density ρ_{crit} is 68 veh/km (23 veh/lane/km), jam density ρ_{jam} is 340 veh/km (113 veh/lane/km), and the capacity drop parameter λ is 0.9. Other ACTM-related parameters such as θ = 0, η = 0.16 are both set as their typical values selected from (Gomes and Horowitz 2006). Similar to the hypothetical cases, the simulation interval T_s for ACTM is set as 15s to guarantee the CFL condition ($T_s \cdot v <$ 500 meters, the minimum cell length shown in Figure 7.2).

Because of the data limitation in JTDB, the location-specific fundamental diagrams cannot be obtained (there are no location-specific densities or speed data available). It is assumed that all locations have the same fundamental diagram calibrated in Figure 7.7. Another problem about the

available data is that these data are aggregated with 15-minute intervals, which cannot capture the flow changes within a short time. The ideal data should be with 1 or 5 minute intervals (Dervisoglu et al. 2009, Chow and Li 2014). Although these limitations exist in the current work, the effectiveness test in this study shows that the selected parameters work well within the test period (peak hours). Thus, the parameters presented in this section are acceptable for this study. The detailed effectiveness test of these parameters is presented in the next section.

7.3 Effectiveness Test for ACTM

7.3.1 Data description

The detector data collected from TRADS are traffic flow data at different locations of the network. These data were selected from May 2011 to August 2011 with 15-minute intervals (the same period as JTDB data used in Section 7.2). The averaged midweek data are used to test the effectiveness of ACTM here and test RAS in the next section, as Wednesday has the heaviest traffic load among all week days.

Detectors are located on the motorway mainline (with a spacing of 500 meters) and different ramps (both on- and off-ramps), from which the demand flows at different origins and the split flows (flows exiting the motorway from off-ramps) at different off-ramps can be extracted. Specifically, demand flows from O, O1, O2, O3, and O4 can be collected from detectors 30030314, 30032404, 30032418, 134 and 30020764, while split flows to D1, D2 and D3 are observed from detectors 30020741, 30032428, and 30032414 (as shown in Figure 7.2). There was no data record for off-ramp 4 during the data collection period, thus, the difference between data collected upstream (30022010) and downstream (30030763) of off-ramp 4 is used as the split flow to D4. Split ratios of four off-ramps can

be calculated by dividing upstream flows of off-ramps (collected from 30030314, 30032408, 30032392 and 30030763) by their corresponding split flows (collected from 30020741, 30032428, 30032414 and the difference between 30022010 and 30030763).

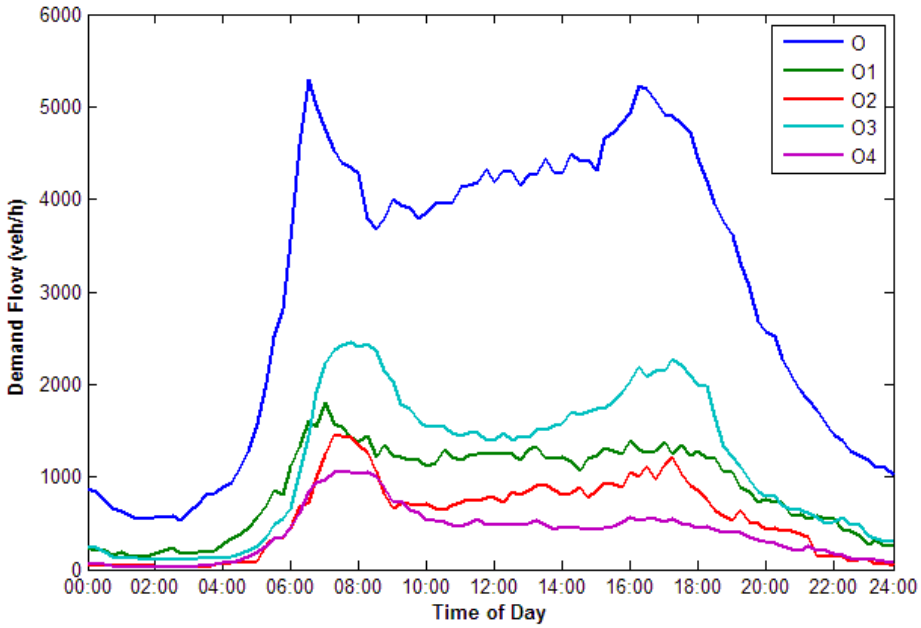


Figure 7.8: Observed demand flows

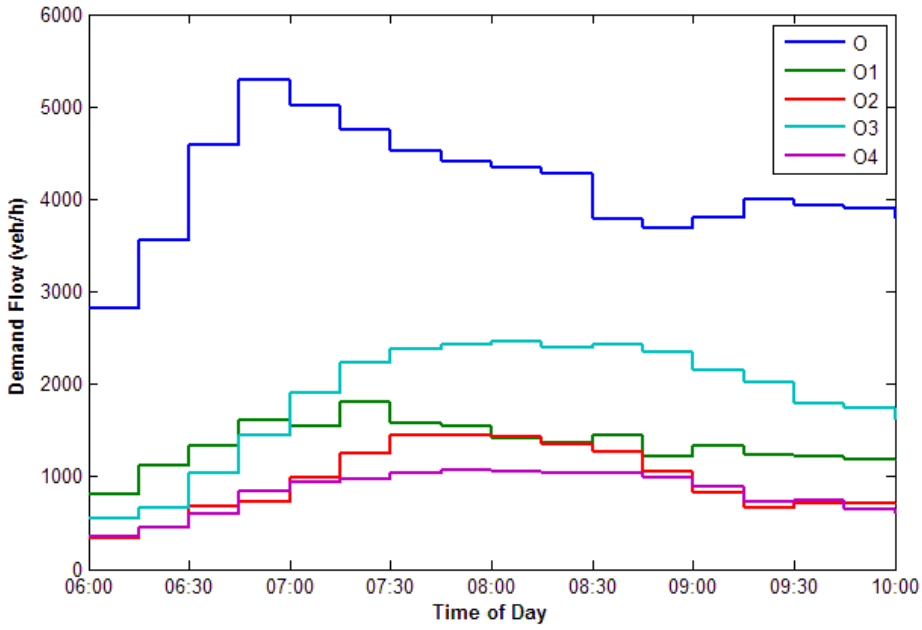


Figure 7.9: Demand flows of AM peak period

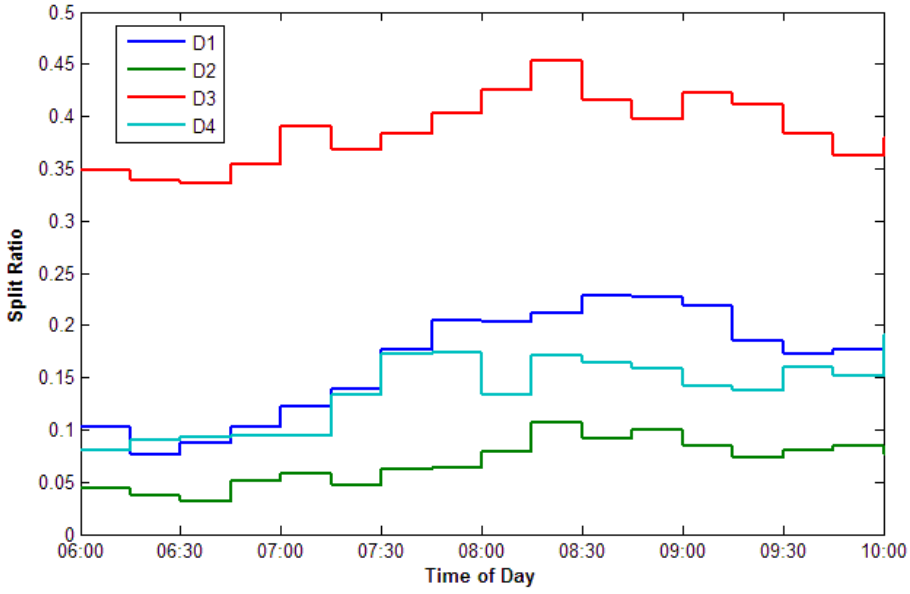
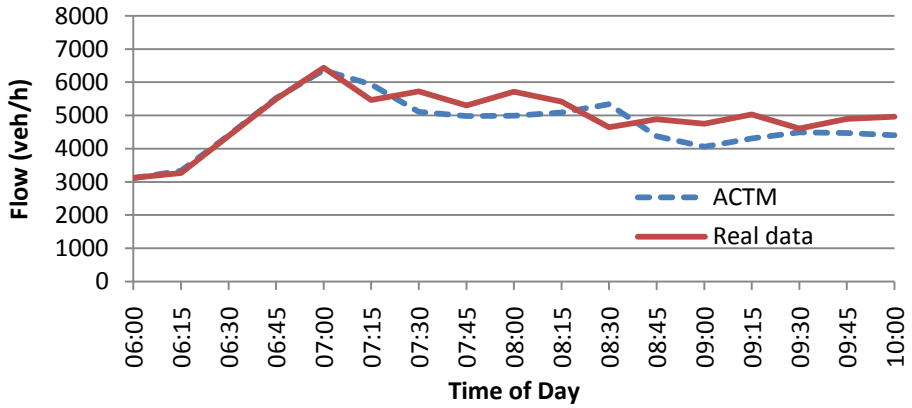


Figure 7.10: Split ratios of AM peak period

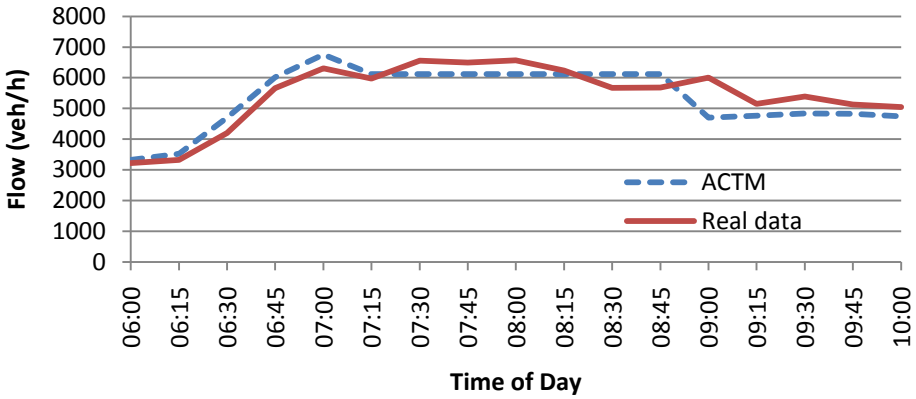
From Figure 7.8, it can be seen that two peak periods including AM peak period (around 06:00-10:00) and PM peak period (around 16:00-20:00) existed during the daily traffic operation. Compared with PM peak period, a heavier traffic load can be observed in AM peak period which will be selected for this study. Demand flows and split ratios between 06:00 and 10:00 are shown in Figures 7.9 and 7.10, which will be input to ACTM at each time step.

7.3.2 Test results

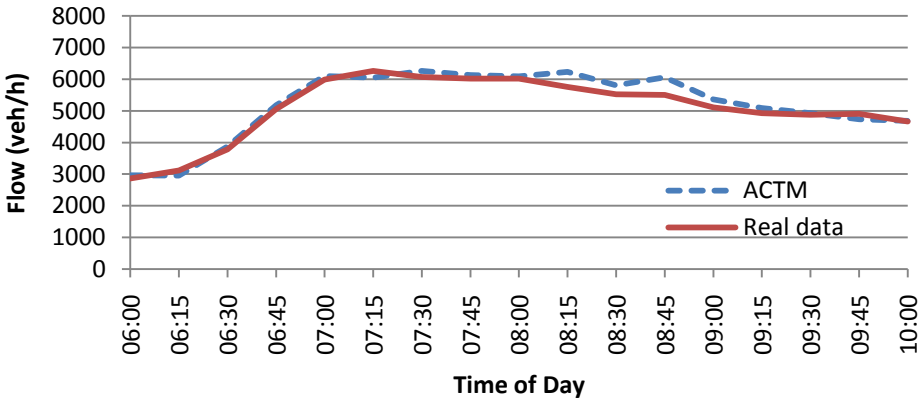
The flow data of three locations at the end of cell 5, cell 9 and cell 18 near three metered on-ramps are used to test the effectiveness of ACTM on simulating real traffic. These three locations were selected because both free-flow states and congestion states can be observed from them. The real flow data at the end of cell 5 and cell 9 were collected from detectors 30032408 and 30032392 respectively. For cell 18, this flow can be obtained by summing data observed from 30030763 and 30020764. The comparison of real observed data and simulated data by ACTM can be seen from Figure 7.11.



(a)



(b)



(c)

Figure 7.11: Comparison of observed and simulated flows

Following (Muñoz et al. 2003) and (Muñoz et al. 2004), the mean-percentage errors (MPE) between observed data and corresponding simulated data can be used to measure the effectiveness of ACTM. The MPE of each cell i ($E_{MPE,i}$) can be calculated by:

$$E_{MPE,i} = \frac{1}{N_k} \cdot \sum_{k=0}^{N_k-1} \left| \frac{q_{out,i}^k - \hat{q}_{out,i}^k}{\hat{q}_{out,i}^k} \right| \cdot 100\% \quad (7.1)$$

where $q_{out,i}^k$ is the cell outflow generated by ACTM, $\hat{q}_{out,i}^k$ is the real observed flow at the same location, N_k is the number of time steps. For three test cells, the calculated MPEs following Equation (7.1) are: $E_{MPE,5}$ =7.4%, $E_{MPE,9}$ =7.4% and $E_{MPE,18}$ =3.6%. The mean error of ACTM at three locations is 6.1%. Thus, ACTM can provide a good simulation of the test network with errors lower than 10%. This low percentage error also indicates that the test motorway network was not well controlled during the data collection period, as observed flows were very close to flows generated by ACTM without control.

7.4 RAS Test

7.4.1 Non-controlled situation

Figure 7.12 illustrates the density evolution without control, where traffic congestion starts forming at Junction 9 (distance around 5 km) around 06:30 in the morning because of increased demand flows from both the mainline and on-ramps. This congestion increases during the next two hours which propagates upstream and worsens the traffic operation at Junction 10 (distance around 2 km). When high demand flows from O1 and O2 arrive between 07:00 and 09:00, severe congestion occurs at Junctions 10 and 9 with the highest density about 70 veh/lane/km, while traffic operation at Junction 7 (distance around 9 km) is smoother with only slight congestion corresponding to the highest density around 30 veh/lane/km. After 08:30, with decreased demand flows, traffic congestion dissipates quickly and traffic operation in the whole network goes back to the free-flow state with mainline density below 20 veh/lane/km.

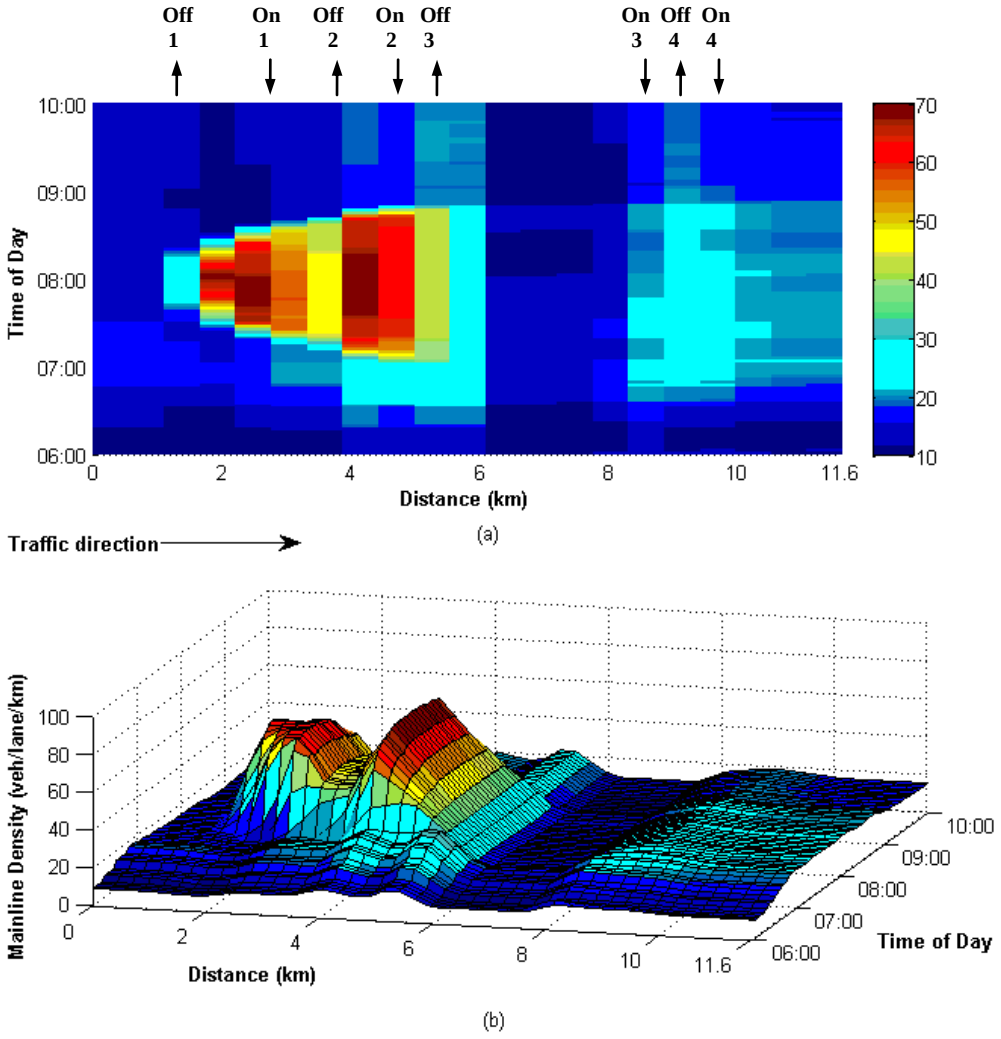


Figure 7.12: Density evolution without control (real network)

7.4.2 Under control of RAS

In contrast to the hypothetical cases, on-ramps in the real network have two lanes. Thus, an even number of vehicles should be released within each control interval. The discrete metering rate set for the real network can be defined as: $C_I = \{2,4,6,8,10,12,14,16,18\}$ veh/ T_c ranging from 240 to 2160 veh/h. The control interval T_c is set as a typical value of 30s (the same as the hypothetical cases). Because the demand flows and parameters of ACTM have been changed in the real-network case, the state set needs to be regulated here following the same method introduced in Section 6.1.2.

(1) $|S_{nmain,I}|$=12 ($n^{up}_{main,i}$= $\rho_{jam,i} \cdot l_i$ = 170 veh, $n^{low}_{main,i}$=0 veh, $\Delta n_{main,i}$=17 veh)

(2) $|S_{qin,I}|$=21 ($q^{up}_{in,i}$ = $q_{cap,i}$ = 6800 veh/h, $q^{low}_{in,i}$= 3000 veh/h, $\Delta q_{in,i}$=200 veh/h)

(3) $|S_{non,I}|$=22 ($n^{up}_{on,i}$= 200 veh, $n^{low}_{on,i}$=0 veh, $\Delta n_{on,i}$=10 veh)

(4) $|S_{don,I}|$=12 ($d^{up}_{on,i}$= 2400 veh/h, $d^{low}_{on,i}$=400 veh/h , $\Delta d_{on,i}$= 200 veh/h)

There are total $|S_I|$ = 12×21×22×12 = 66528 states in the state set. Other parameters for RAS and RAS-EQ including three learning parameters in the real-network case are the same as the hypothetical cases introduced in Chapter 6. RAS takes about 500000 episodes (about 1 hour) to get convergence in the real-network case (see Figure 7.13). From Figure 7.14, it can be seen that RAS can greatly alleviate traffic congestion in locations between 1 and 6 km (around on-ramp 1 and 2), but cannot completely eliminate it with the minimum metering rate 240 veh/h. The highest mainline density is around 25 veh/lane/km which is slightly higher than the critical value (23 veh/lane/km). In the location between 8 and 10 km (around on-ramp 4), mainline density has no obvious changes before 08:00 compared with the non-controlled situation and keeps between 25 and 30 veh/lane/km. After 08:00 (as outlined in red), traffic congestion can be eliminated with the density below the critical value.

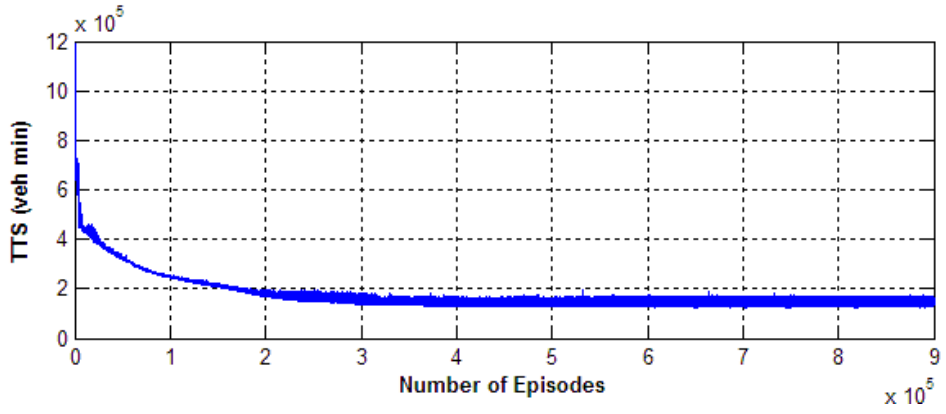


Figure 7.13: RAS convergence (real network)

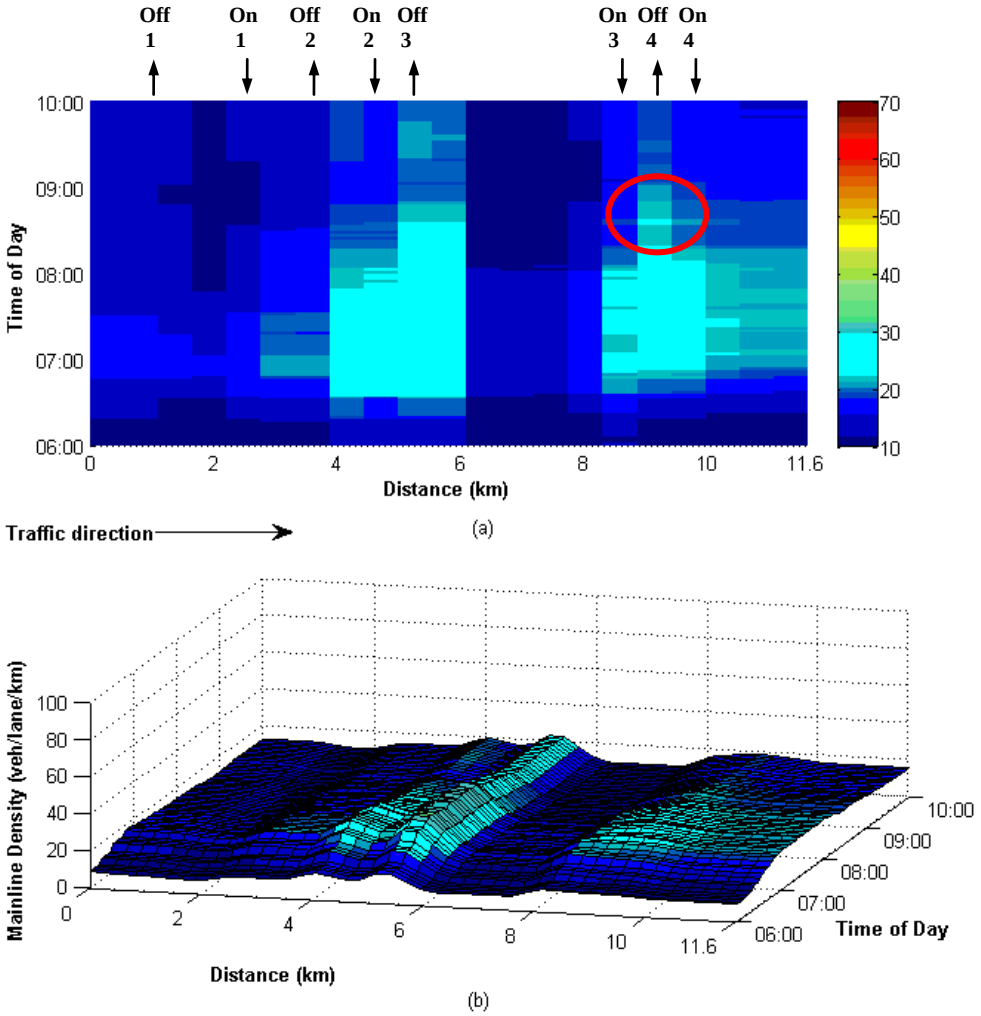


Figure 7.14: Density evolution under RAS (real network)

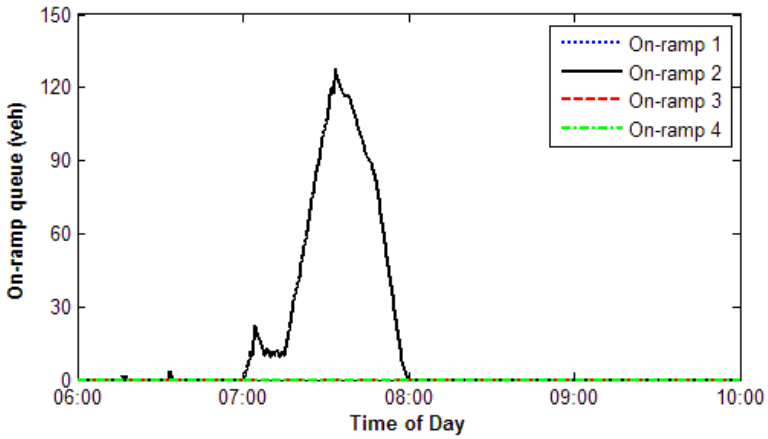


Figure 7.15: On-ramp queue under RAS (real network)

Figure 7.15 shows on-ramp queues under the control of RAS. Among all three controlled on-ramps (on-ramp 1, 2 and 4), only on-ramp 2 is strictly controlled with the longest queue around 120 veh. Both on-ramp 1 and 4 are under loose control with no waiting vehicles at on-ramps. On-ramp 3 is not controlled, and there is no queue on this on-ramp. In this situation, the network TTS can be reduced to 125299 veh.min (an 18.5% reduction from the non-controlled situation as illustrated in Table 7.1).

7.4.3 Under control of RAS-EQ

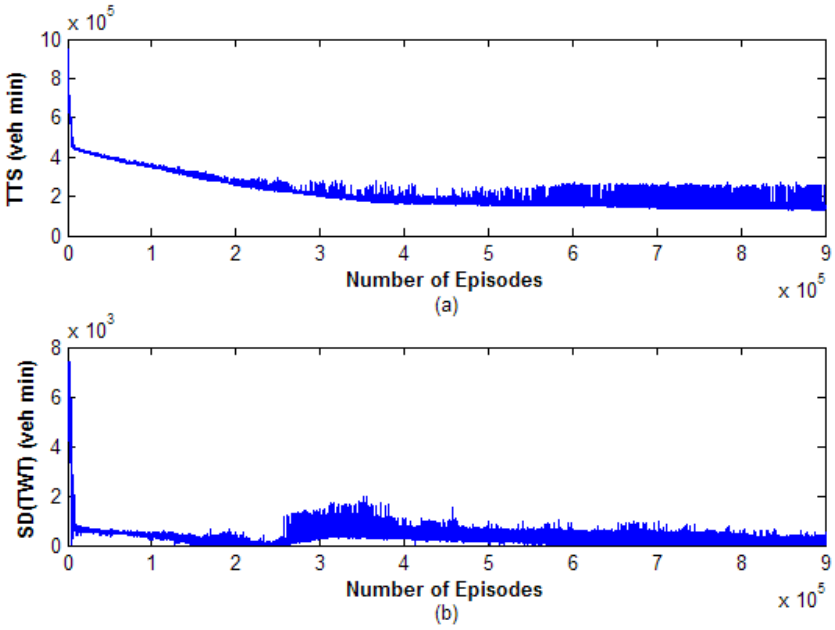


Figure 7.16: RAS-EQ convergence (real network)

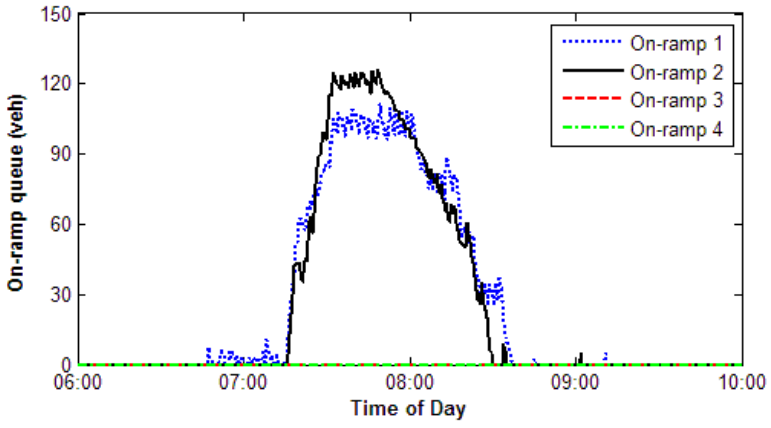


Figure 7.17: On-ramp queue under RAS-EQ (real network)

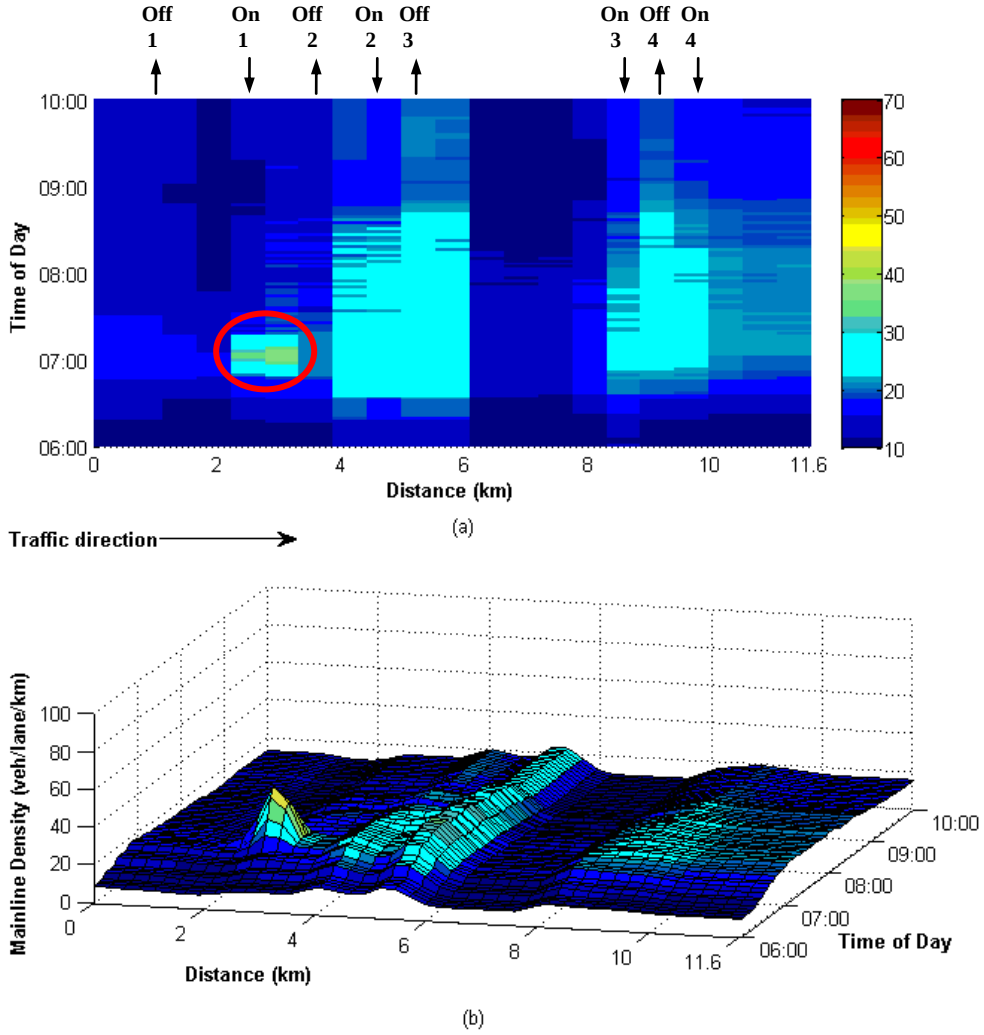


Figure 7.18: Density evolution under RAS-EQ (real network)

This test will focus on the equity issue of on-ramps 1 and 2, as these two on-ramps are closely located and have a great effect on each other. On-ramp 4 is located far from on-ramp 2 (with a distance about 5 km), which only causes slight congestion and has no obvious impacts on its upstream traffic. Therefore, on-ramp 4 is not involved in the equity test.

It has been shown in the multi-ramp case (Section 6.2) that, higher δ_2 leads to higher equity. To obtain a high importance on equity, the weight value δ_2 of ramp agents 1 and 2 (controlling on-ramp 1 and 2 respectively) is set as 0.9, while for ramp agent 4 (controlling on-ramp 4) δ_2 is set as 0, which

means equity is not considered by ramp agent 4. The influence of different weight values has been tested in Section 6.2 which will not be repeated here. This section only focuses on RAS-EQ with a high equity (δ_2 = 0.9). Following the same way mentioned in Section 6.2.5, the value of $SD_{ef}(TWT)$ is set as 15.9 veh.min according to Figure 7.21. The RAS-EQ convergence can be seen from Figure 7.16, which takes about 700000 episodes (75 mins) to find the acceptable solution.

Figure 7.17 illustrates vehicle queues under RAS-EQ. Because on-ramp 4 is not involved in the equity consideration, both on-ramps 3 and 4 have no waiting vehicles. On the other hand, vehicle queues of on-ramps 1 and 2 can be kept at almost the same level and thus lead to a more equal distribution of TWT at these two on-ramps. However, to maintain this equity, a long vehicle queue builds up at on-ramp 1 and causes unnecessary congestion on the mainline around 07:00 (as outlined in Figure 7.18). Both the on-ramp queue and mainline congestion lead to a higher network TTS (135255 veh.min) than RAS.

7.4.4 Comparison with ALINEA

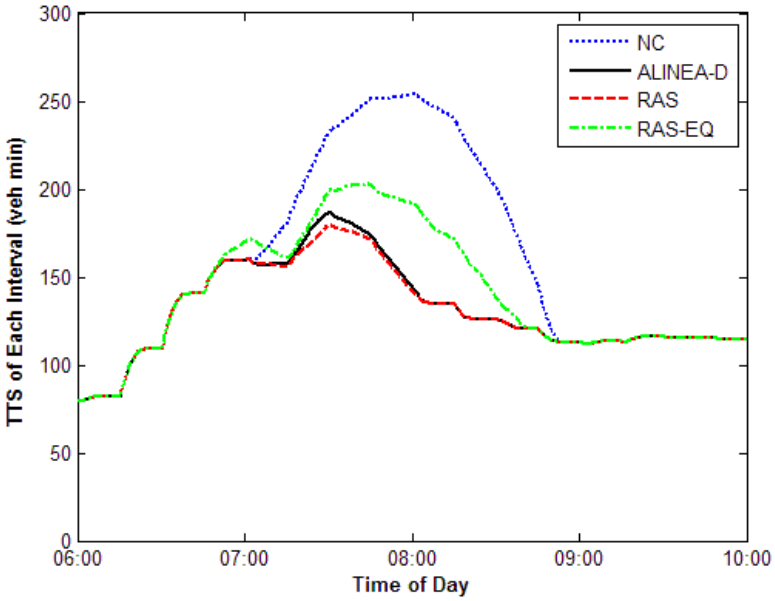


Figure 7.19: TTS comparison (real network)

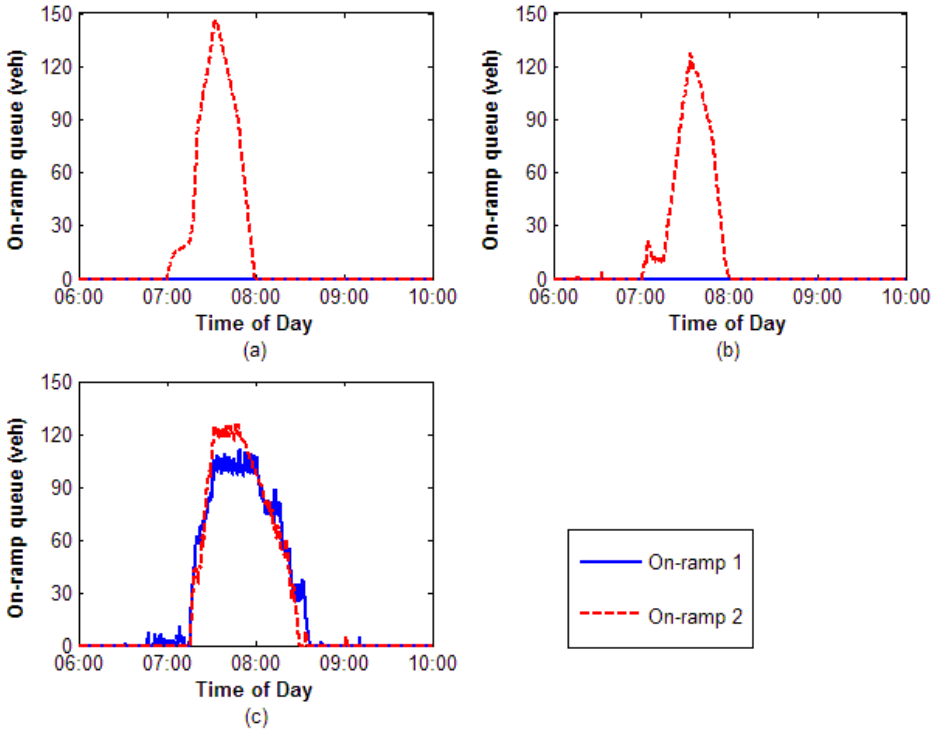


Figure 7.20: On-ramp queue comparison (real network): (a) ALINEA-D, (b) RAS, (c) RAS-EQ

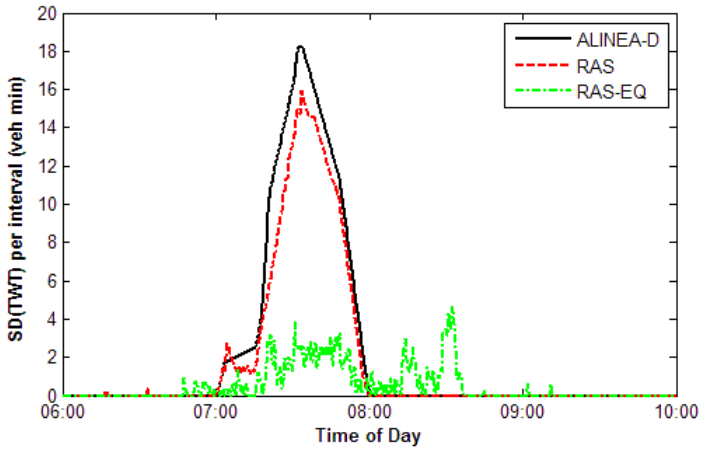


Figure 7.21: SD (TWT) comparison (real network)

In the real-network case, the practical ALINEA strategy ALINEA-D with integer number of vehicles is used for analysis. Within the same rate range of RAS, control action generated by ALINEA-D at each step is rounded to the nearest even number (for a two-lane on-ramp in the real network). The

parameters calibrated from Section 6.1 also work well in the real-network case and are accordingly used in this test.

Figure 7.19 illustrates the TTS comparison of three strategies. Although RAS is slightly better than ALINEA-D between 07:30 and 08:00 in terms of TTS reduction, the overall performance of RAS and ALINEA-D during the whole test period is very close, with respectively 17.9% (126200 veh.min) and 18.5% (126200 veh.min) reduction on the network TTS. RAS-EQ can reduce TTS by 12% (135255 veh.min) which is less than the other two strategies.

An on-ramp queue comparison is illustrated in Figures 7.20, from which it can be seen that queue difference between two on-ramps under RAS-EQ is much smaller than RAS and ALINEA-D. Accordingly, as shown in Figure 7.21, RAS-EQ can keep a much lower SD (TWT) for most of the time (especially between 07:00 and 08:00) than both RAS and ALINEA-D. As illustrated in Table 7.1, the overall SD (TWT) of the whole test period can be kept at a low level about 5.7 veh.min by RAS-EQ. Without considering equity, RAS has a much higher SD (TWT) than RAS-EQ which is 1710.2 veh.min. Among the three strategies, ALINEA-D has the worst performance on maintaining equity, which has the highest SD (TWT) about 2105.8 veh.min.

Table 7.1: TTS and SD(TWT) under different strategies (real network)

Strategies	NC	ALINEA-D	RAS	RAS-EQ
TTS (veh.min)	153657	126200	125299	135255
TTS reduction (%)	–	17.9	18.5	12.0
SD (TWT) (veh.min)	0	2105.8	1710.2	5.7

7.5 Summary and Discussion

In this chapter, a real motorway network selected from the M6 motorway in the UK was applied to test the performance of RAS on dealing with real observed traffic data. Specifically, two kinds of data collected from the

highways agency database JTDB and TRADS were used to set parameters for ACTM, based on which simulation experiments were designed to test RAS. The main difference between hypothetical cases and the real-network case is the demand profiles (for the motorway mainline and on-ramps) and the split ratios (for off-ramps) used for simulation. In the hypothetical cases presented in Chapter 6, the simplified trapezoidal demands and fixed split ratios were adopted. In the real-network case, on the other hand, both demand flows and split ratios were derived from the practical traffic data which fluctuated greatly during the test period (see Figure 7.9 and 7.10). Although traffic conditions fluctuated in the real-network case, RAS could still achieve a good performance on improving efficiency and maintaining equity. For efficiency improvement, RAS reduced the network TTS by 18.5% which was slightly better than ALINEA (17.9%). For maintaining equity, RAS-EQ could keep a much lower SD(TWT) than ALINEA (2105.8 veh.min), which was around 5.7 veh.min.

As previously mentioned in Chapter 4, a self-learning system needs to be trained by a simulation model before it can be used for real traffic (Jacob and Abdulhai 2010, El-Tantawy et al. 2013). In the tests presented in Chapters 6 and 7 (using ACTM), RAS usually needs 5 to 70 minutes to get convergence according to different network scopes (from 4 km to 12 km) and test periods (from 1 h to 4 h). Although ALINEA can be used immediately for the real traffic control, a calibration process (before real applications) is required by this system to guarantee the algorithm performance. When ACTM is used, 10 to 60 minutes are required to calibrate ALINEA in different scenarios. Therefore, the learning time required by RAS is reasonable according to the tests in this study.

CHAPTER 8 EXTENSION TO CONGESTION CASE

The case studies presented in Chapters 6 and 7 showed the effectiveness of RAS in dealing with congestion in peak hours. This kind of congestion is usually named recurrent congestion, because it is caused by the daily traffic operation with increased demands in peak hours (Skabardonis et al. 2003). Considering the daily normal traffic operation on motorways, most of existing ramp metering systems focused on the recurrent congestion which is also the main concern of this research. However, besides the recurrent congestion, there is the other traffic congestion named non-recurrent congestion that also causes delays for motorway users. Instead of the normal traffic operation, the non-recurrent congestion is caused by incidents (e.g. vehicle collisions, breakdowns, spilled loads), inclement weather or other unexpected events on motorways (Dowling et al. 2004).

Non-recurrent congestion in the literature usually refers to incident-induced congestion (Hall 1993, Skabardonis et al. 2003), as incidents are the main cause of this kind of congestion. Compared with recurrent congestion, non-recurrent congestion is more uncertain (e.g. uncertain capacity during the incident), which may interrupt the normal traffic operation and complicate the control process. In this chapter, one attempt has been done to extend the original RAS to deal with incident-induced non-recurrent congestion. Some simulation-based experiments are designed to test the effectiveness of RAS in an incident situation.

As mentioned in Chapter 5, RAS is independent from ACTM and it can also be reused by other traffic simulation models. In this chapter, the other purpose is to test the reusability of RAS using the traffic simulation software

AIMSUN[9]. This test provides an example about how to link RAS-related files developed in Chapter 5 with other commercial software (such as AIMSUN).

The organisation of this chapter is as follows. Section 8.1 briefly introduces the existing ramp metering strategies dealing with incidents and their limitations. Section 8.2 presents the influence of incidents on the traffic flow operation with uncertain capacity. The design of extended RAS including its structure and algorithm is given in Section 8.3. Then the simulation experiments based on AIMSUN and relevant results are discussed in Section 8.4. Finally, Section 8.5 gives some conclusions and the summary of this chapter.

8.1 Related Work

Non-recurrent congestion caused by incidents is a main cause of traffic delays on motorways. Studies have shown that incident-induced delays account for more than 60% of all the delays on some motorway networks (Prevedouros et al. 2008). To alleviate incident-induced congestion, traffic incident management systems (TIMS) with advanced traffic control technologies such as ramp control, variable message signs (VMS) and adaptive arterial signal control have been developed (Ozbay and Kachroo 1999). As an important part of these TIMSs, incident-responsive ramp metering control has been proposed and also studied in some literature.

As an early attempt, Greenlee and Payne (Greenlee and Payne 1977) proposed a system structure for solving the incident-responsive ramp metering problem based on a simple macroscopic flow model, but no computational solution was provided. Considering some dynamic incident

---

9 The AIMSUN used in this study is the version 6.1. Detailed information about this software can be found in http://www.aimsun.com/wp/.

features, such as incident duration and traffic arrival rates, Wang (Wang 1994) formulated a more complex problem and solved it using a linear programming model. In the study introduced in (Jiuh-Biing and Mei-Shiang 2007), lane-changing and queuing behaviour caused by an incident was modelled. A stochastic optimal control method was proposed to solve this problem. These strategies are all model-based methods, as they all need predefined models to generate control actions. In addition to model-based methods, a model-free approach was recently proposed in (Jacob and Abdulhai 2010). As introduced in Chapter 2, this system is based on RL and combined with VMS to deal with incident control problems. In this chapter, this RL-based system can be named as a DRL (direct reinforcement learning) method, as it is based on the basic Q-learning and works without the help of models. Thus, the RAS developed in Chapter 4 is also a DRL system.

Both model-based and model-free strategies mentioned above have advantages and limitations. The model-based method is based on accurate models of the controlled road traffic. These models are used to predict traffic states or specify control rules for traffic control. This method is thus theoretically reliable and can be used immediately once defined. Nonetheless, models cannot continuously learn to improve themselves and thus have poor adaptability (Jacob and Abdulhai 2010). A model-free method such as DRL, on the other hand, can continuously learn from interactions with road traffic without using models. However, DRL can only learn from real interactions with the traffic flow operation and cannot make full use of historical data (traffic information that has been collected). Because of this limitation, DRL usually needs a great number of trials to obtain the best control strategy for highly uncertain problems, such as incident-responsive ramp metering. Fortunately, in the practical incident management, some useful information such as the distribution of road

capacity can be derived from the historical database and used to build related models (Smith et al. 2003). This provides an opportunity to combine model-based and model-free methods to achieve benefits from both sides. For this purpose, an indirect reinforcement learning (IRL) approach based on Dyna-Q architecture has been developed and outlined here. The IRL algorithm developed in this study is extended from the basic RAS, which has three features: (1) Similar to the model-free method (the original RAS), IRL can continuously learn from real interactions with the road traffic. (2) Similar to the model-based method, some simple models are maintained to speed up the learning process. (3) Another distinguishing feature of IRL is that model can be improved based on the observed traffic data (from detectors).

8.2 Influence of Incident

Before the detailed design of an IRL system, some background knowledge related to incident-induced traffic operation is presented here.

8.2.1 Traffic operation during the incident

Usually, incidents on motorways refer to any non-recurrent events that cause a rapid reduction of the motorway capacity such as vehicle collisions, breakdowns and spill loads (Farradyne 2000). Figure 8.1 illustrates a typical incident situation on a two-lane motorway. When an incident happens, one or more lanes of the motorway will be blocked according to the incident extent. The outflow of the incident section (q_{out}) will be affected until this incident is cleared and traffic operation returns to its normal situation.

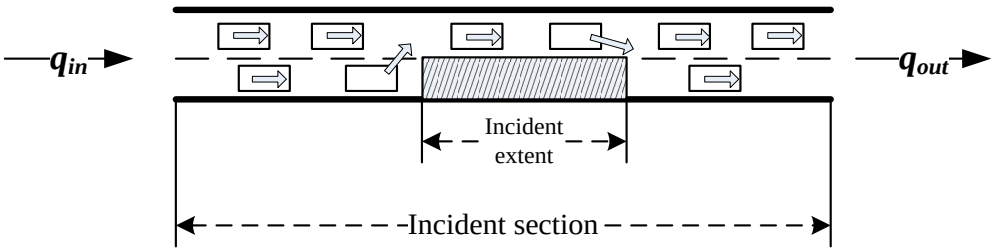


Figure 8.1: A typical incident situation

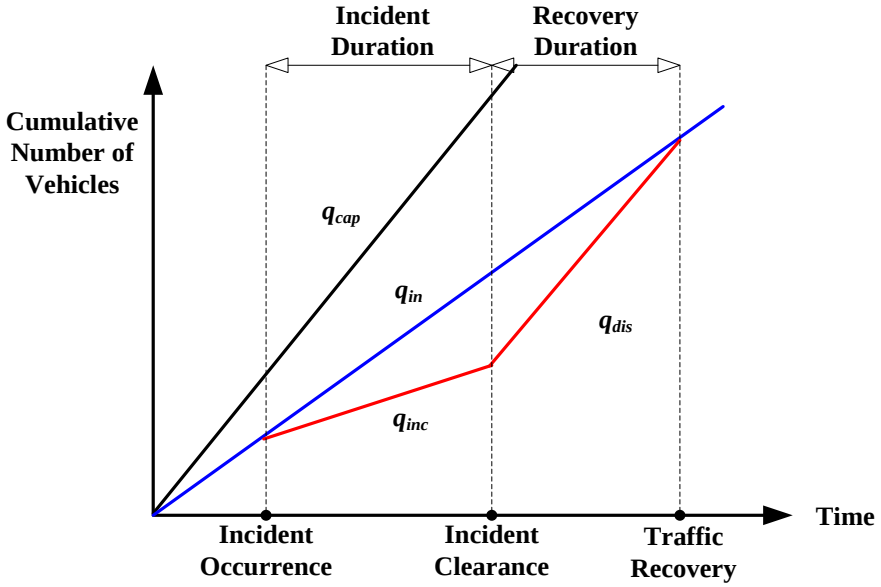


Figure 8.2: Traffic operation under incidents

The traffic operation during an incident is usually presented by a simple deterministic queuing diagram shown in Figure 8.2 (Fu and Rilett 1997, Li et al. 2006, Wang 1994). The slope of each line represents the flow rate. When an incident occurs, the road capacity q_{cap} will be reduced to the incident capacity q_{inc} because of the lane blockage. The outflow of incident section during the whole incident duration (between incident occurrence and incident clearance) is restricted by q_{inc} . After the incident clearance, the vehicle queue on the motorway will gradually dissipate and the traffic flow will recover to its normal situation. During the recovery, the outflow of incident section equals the queue discharge rate q_{dis} which is the same as original road capacity. Therefore, when incident-induced congestion occurs, q_{out} can

be represented by two flows, i.e. q_{inc} (from incident occurrence to clearance) and q_{dis} (from incident clearance to recovery).

8.2.2 Uncertain capacity

The deterministic queuing diagram gives a general expression of the traffic operation in an incident situation where the incident capacity (q_{inc}) determines traffic flow during an incident.

In the practical incident situations, many uncertain factors such as incident locations, road conditions, and driving behaviour have great effects on the incident capacity. Thus, this parameter is usually not constant and has been considered as a stochastic value in many studies such as (Fu and Rilett 1997, Li et al. 2006, Smith et al. 2003). In these studies, the uncertainty of incident capacity is modelled by the capacity reduction [10] which is the difference between the original capacity (q_{cap}) and incident capacity (q_{inc}). In this chapter, the uncertain capacity during the incident will be considered in the IRL system.

8.2.3 Simulating uncertain capacity

Considering the lack of direct functions in AIMSUN for simulating stochastic capacity reduction during the incident, an alternative approach as proposed by (Hadi et al. 2007) can be used. In their work, a relationship between the vehicle speed in the incident section and road capacity reduction was found in the AIMSUN's simulation model (see Figure 8.3). Road capacity can be adjusted through changing the permitted vehicle speed in the incident

---

[10] The capacity reduction is not the same as capacity drop phenomenon introduced in Chapter 2. The capacity reduction is caused by incident-induced lane blockage on motorways. Therefore, it cannot be eliminated until the incident is cleared. On the other hand, the capacity drop can be avoided through keeping mainline density below the critical value.

section. To simulate stochastic capacity reduction, the permitted speed in the incident section can be regulated to make the capacity reduction follow a predefined distribution.

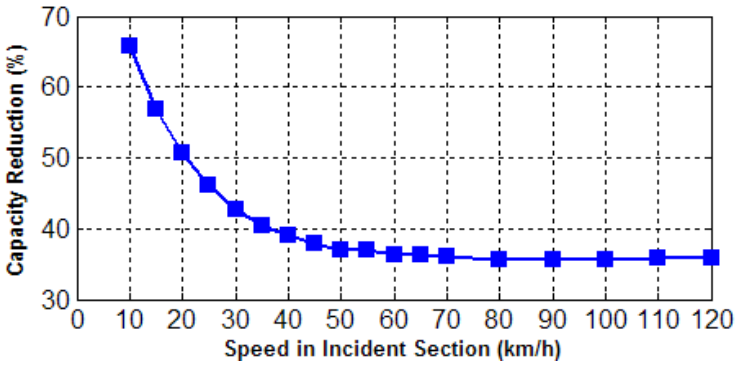


Figure 8.3: Relationship between vehicle speed and capacity reduction

Many potential distributions can be used to model capacity reduction during an incident, such as the normal distribution, beta distribution and Johnson distribution (Smith et al. 2003). For a three-lane motorway, an incident with one lane blocked usually reduces road capacity by around 50% (Hadi et al. 2007). To simplify the problem, it is assumed in this chapter that the capacity reduction follows a normal distribution with mean 50% and standard deviation 5% for one-lane blocked incident on a three-lane motorway (Figure 8.4.). Only one-lane blocked incident is considered in this study, as it takes the largest part of all lane-blocking incidents (Rodgers et al. 2006).

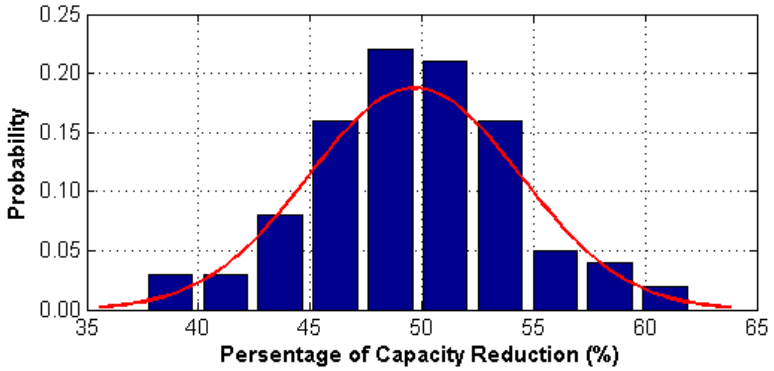


Figure 8.4: Histogram for capacity reduction distribution

8.3 IRL Strategy

After a description of traffic operation during the incident, this section will focus on the development of an IRL system. The IRL system was extended from the basic RAS and following the Dyna-Q architecture.

8.3.1 Dyna-Q architecture

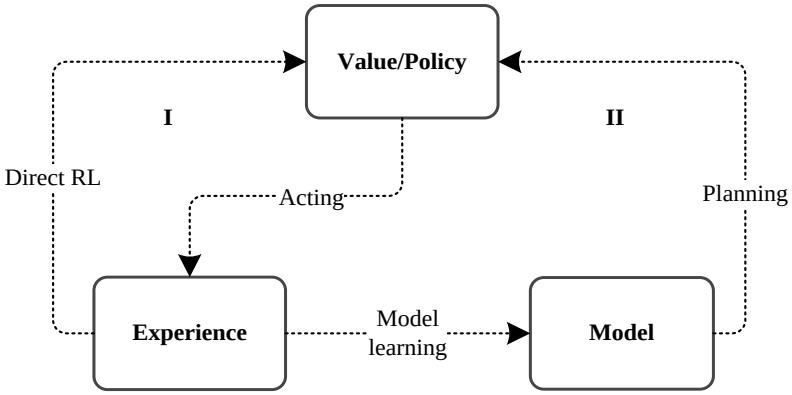


Figure 8.5: Dyna-Q architecture

Dyna-Q architecture is an extension of standard Q-learning that integrates planning, acting and learning together (Sutton 1991). Unlike Q-learning which learns from the real experience without a model, Dyna-Q learns a model and use this model to guide the agent (Kaelbling et al. 1996). After capturing the real experience, two loops run to learn the optimal actions that can help the agent obtain the maximum Q value in Dyna-Q architecture (see Figure 8.5).

In Loop I, direct RL is the standard Q-learning process that can be used to interact with the real external environment. Loop II contains two main tasks: (1) model learning is used to improve the model accuracy through obtaining new knowledge from real experience, (2) planning is the same process of direct RL except that it uses the experience generated by a model. Acting is the action execution process.

Applying a model, the agent can predict reactions of its external environment before executing a specific action, which provides an opportunity for agent to update Q value before receiving the real feedback. Simultaneously, direct RL is running to update the Q value through the real interaction. Therefore, optimal policy is learned through both real experience and predictions. By using this strategy Dyna-Q can learn faster than Q-learning in many situations (Sutton and Barto 1998).

8.3.2 IRL structure

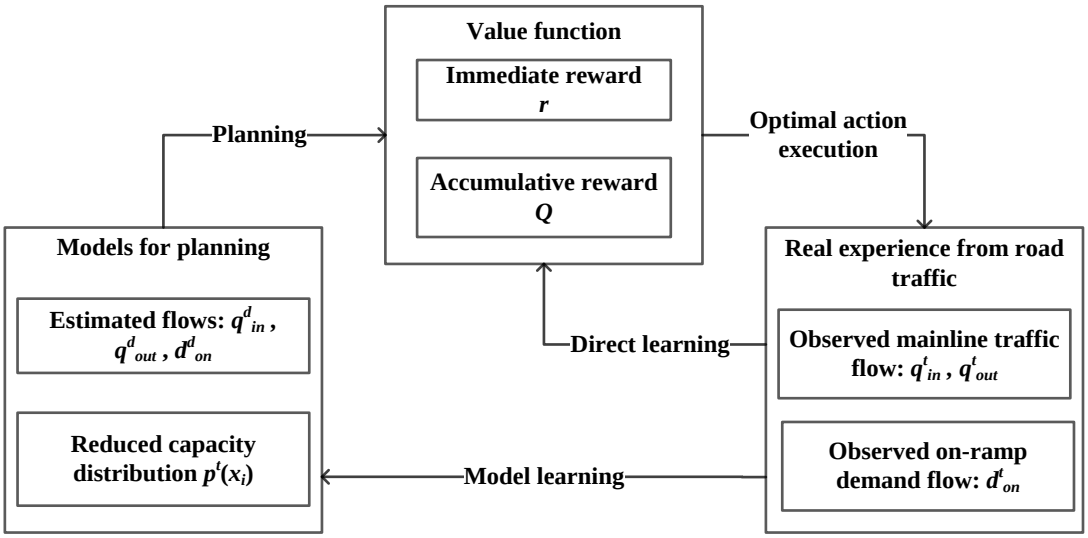


Figure 8.6: IRL structure

Following Dyna-Q architecture, the structure of IRL agent with three components is shown in Figure 8.6. Real experience, composed of observed flows from the mainline (q^t_{in} and q^t_{out}) and on-ramp (d^t_{on}), is used to update models and Q values at control step t. Two simple models of the estimated traffic flows (q^d_{in}, q^d_{out} and d^d_{on}) and road capacity reduction ($p^t(x_i)$) are updated continuously to generate simulated experience at planning step d. Based on models, several planning steps are triggered to generate reward and update Q values before real experience is captured. To make these

three components work, three basic elements (state, action and reward) and two models (reduced capacity and estimated flows) are defined as follows.

Reward

In this chapter, the ramp agent is tested in a simple network with only one on-ramp. It has been shown in Section 2.2.1 (mechanism (1)) that, without off-ramps, ramp metering can reduce the total time spent on the whole motorway network through avoiding capacity drop. However, when the incident happens, the capacity cannot recover to its original value until this incident is cleared. In this case, the primary objective, i.e. reducing the total time spent on the network is invalid. As the mainline traffic is more important than the on-ramp traffic, a similar objective proposed by (Wang, 1994) is adopted here. Specifically, the IRL algorithm is aiming to transfer delays from the motorway mainline to the on-ramp (reducing the mainline travel time only) without making the on-ramp queue exceed the on-ramp storage space and worsening other environmental problems such as improving vehicle emissions. Since delays in the congestion situation are mainly caused by vehicle queues, the objective considered in this chapter can be achieved through balancing the vehicle queue length on the mainline and on-ramp. For this purpose, a new reward can be defined as follows:

$$r^t = \begin{cases} -\xi \cdot \dfrac{\hat{n}_{main}^t - \hat{n}_{main}^{\min}}{\hat{n}_{main}^{\max} - \hat{n}_{main}^{\min}} - (1-\xi) \cdot \dfrac{\hat{n}_{on}^t - \hat{n}_{on}^{\min}}{\hat{n}_{on}^{\max} - \hat{n}_{on}^{\min}}, \\ \qquad\qquad \text{if } n_{main}^t < n_{main}^{\max} \text{ and } n_{on}^t < n_{on}^{\max} \\ -1, \quad \text{otherwise} \end{cases} \tag{8.1}$$

$$\begin{cases} \hat{n}_{main}^t = n_{main}^t - n_{main}^{t-1} \\ \hat{n}_{on}^t = n_{on}^t - n_{on}^{t-1} \end{cases} \tag{8.2}$$

$$\begin{cases} n_{main}^t = n_{main}^{t-1} + T_c \cdot (q_{in}^{t-1} + m_r^{t-1} - q_{out}^{t-1}) \\ n_{on}^t = n_{on}^{t-1} + T_c \cdot (d_{on}^{t-1} - m_r^{t-1}) \end{cases} \tag{8.3}$$

where, r^t is the immediate reward at control step t, $\hat{n}_{main}^t$ and $\hat{n}_{on}^t$ is the queue increase on the mainline and on-ramp at step t, respectively. $\hat{n}_{main}^{\max}$ and $\hat{n}_{main}^{\min}$ denote the maximum and minimum mainline queue increases. Similarly, $\hat{n}_{on}^{\max}$ and $\hat{n}_{on}^{\min}$ are the maximum and minimum on-ramp queue increases. These maximum and minimum values are used to normalise queue increases, which can be estimated from the previous observed flow rates of the network. n_{main}^t and n_{on}^t can be obtained according to vehicle conservation. ξ ($\xi \in [0,1]$) is the weight that indicates the importance of traffic on the mainline and on-ramp. Through assigning different weight values, two queues can be properly balanced to achieve the control objective.

State and action

Similar to the recurrent congestion situation, the reward defined for the non-recurrent congestion is also related to n_{main}^t and n_{on}^t. Thus, to satisfy the Markov property, the state set used here is the same as the one defined in Section 4.3.1. The control action is related to a set of discrete metering rates with 9 flow rates between the minimum (2 veh/min) and maximum (18 veh/min) values. The interval between each two rates is 2 veh/min.

Capacity reduction

For the purpose of real application, capacity reduction is discritised into y intervals, which can be represented by a vector $X = \{x_0, x_1, \cdots, x_{y-1}\}$ with y elements. After each real observation at control step t, a simple estimation method shown below can be used to learn and update the probability distribution for x_i $(i = 0,1,,2,...,y-1)$.

$$P^t(x_i) = \frac{m_i' + m_i}{\sum_{j=0}^{y-1}(m_j' + m_j)} \tag{8.4}$$

where, m_i' is the number of samples for x_i observed from previous iterations and steps. m_i ($m_i = 0$ or 1) is the number of samples for x_i estimated in the current time step t. In this study, y is set to 10 with the capacity reduction ranging from 37% to 62% (following the capacity reduction introduced in Section 8.2.3).

Estimated flows

Given a capacity reduction percentage x_i, the estimated mainline outflow in the congested situation at planning step d, q_{out}^d can be calculated according to the queuing diagram (Figure 8.2):

$$q_{out}^d = \begin{cases} q_{cap} \cdot (1 - x_i), & d \le N_d \\ q_{cap}, & \text{otherwise} \end{cases} \tag{8.5}$$

$$N_d = \left\lceil \frac{IncidentDuration}{T_c} \right\rceil \tag{8.6}$$

where, q_{cap} is the original road capacity (veh/min) before the incident, N_d is the number of control steps before incident clearance, $\lceil \ \rceil$ is a ceiling function that can convert the incident duration to a number of control steps according to the control interval T_c. Although some studies (Li et al. 2006, Valenti et al. 2010) mentioned that the incident duration cannot be explicitly estimated in highly uncertain situations, in this study, the incident duration is deterministic and assumed to be known in advance. How to explicitly predict the incident duration is not the focus of this study.

An estimation method described in (Wang 1994) can be used to estimate inflows of the mainline and demand flows of the on-ramp. This method simply average the most recently observed flow data to predict demand flows (for both the mainline and on-ramp) for the next several steps. In the model presented here, the flow data collected from the last N steps (N=5)

are used for the estimation. Therefore, the mainline inflows and on-ramp demand flows in the planning process can be calculated by:

$$q_{in}^{t,t+1} = \frac{\sum_{n=0}^{N-1} q_{in}^{t-n}}{N} \tag{8.7}$$

$$d_{on}^{t,t+1} = \frac{\sum_{n=0}^{N-1} d_{on}^{t-n}}{N} \tag{8.8}$$

where, $q_{in}^{t,t+1}$ and $d_{on}^{t,t+1}$ are estimated mainline inflow and on-ramp demand flow for the planning process between control step t and $t+1$.

8.3.3 IRL algorithm

Based on the Dyna-Q architecture and models described in Sections 7.3.1 and 7.3.2, an IRL algorithm is developed in this section which is extended from the single-objective algorithm of RAS (introduced in Section 4.4.1).

The IRL algorithm is episode-based. Each episode (or iteration) starts from incident occurrence and terminates when the incident is cleared and the traffic flow returns to its normal situation, or the simulation in AIMSUN has been finished (N_t steps from the algorithm is triggered to the simulation in AIMSUN has been finished). $s^{initial}$ is the state before the incident occurrence. If the incident is cleared ($t+1 \geq N_d$) and the traffic state returns to its initial state ($s^t = s^{initial}$), then the traffic flow has recovered to its normal situation. The whole algorithm of IRL is described in Figure 8.7.

One episode of the IRL algorithm contains two loops corresponding to the two loops in Dyna-Q architecture shown in Figure 8.5. Loop I is related to the control step t, and loop II is related to planning step d. After initialisation, the direct learning is triggered. This process is the same as the basic Q-learning process of the single-objective algorithm introduced in

Section 4.4.1. After that, the model learning process is used to update the distribution of capacity reduction $p^t(x_i)$, mainline inflow $q_{in}^{t,t+1}$ and on-ramp demand flow $d_{on}^{t,t+1}$ for planning. Finally, ten planning steps[11] following the control step t are triggered to update Q table according to the estimated flows. Sometimes, the planning process does not need to finish all ten steps. When the flow returns to its normal situation (i.e. $s^t = s^{initial}$ and $d+1 \geq N_d$), the planning process will end immediately.

For implementation, the main difference between the IRL algorithm and single-objective algorithm of RAS is the function ***startStateTransition()***. Besides the basic Q-learning process, model learning and planning are incorporated in the function ***startStateTransition()*** of the IRL algorithm. The source code for this function is shown in Appendix A.2, code 19.

[11] The selection of ten planning steps between each two learning steps in this chapter is for ease of presentation. For real applications, this number is restricted by the control interval (time duration between two learning steps). For example, if the control interval is 30 s, the determination of the number of planning steps should guarantee that all planning steps can be finished within 30 s.

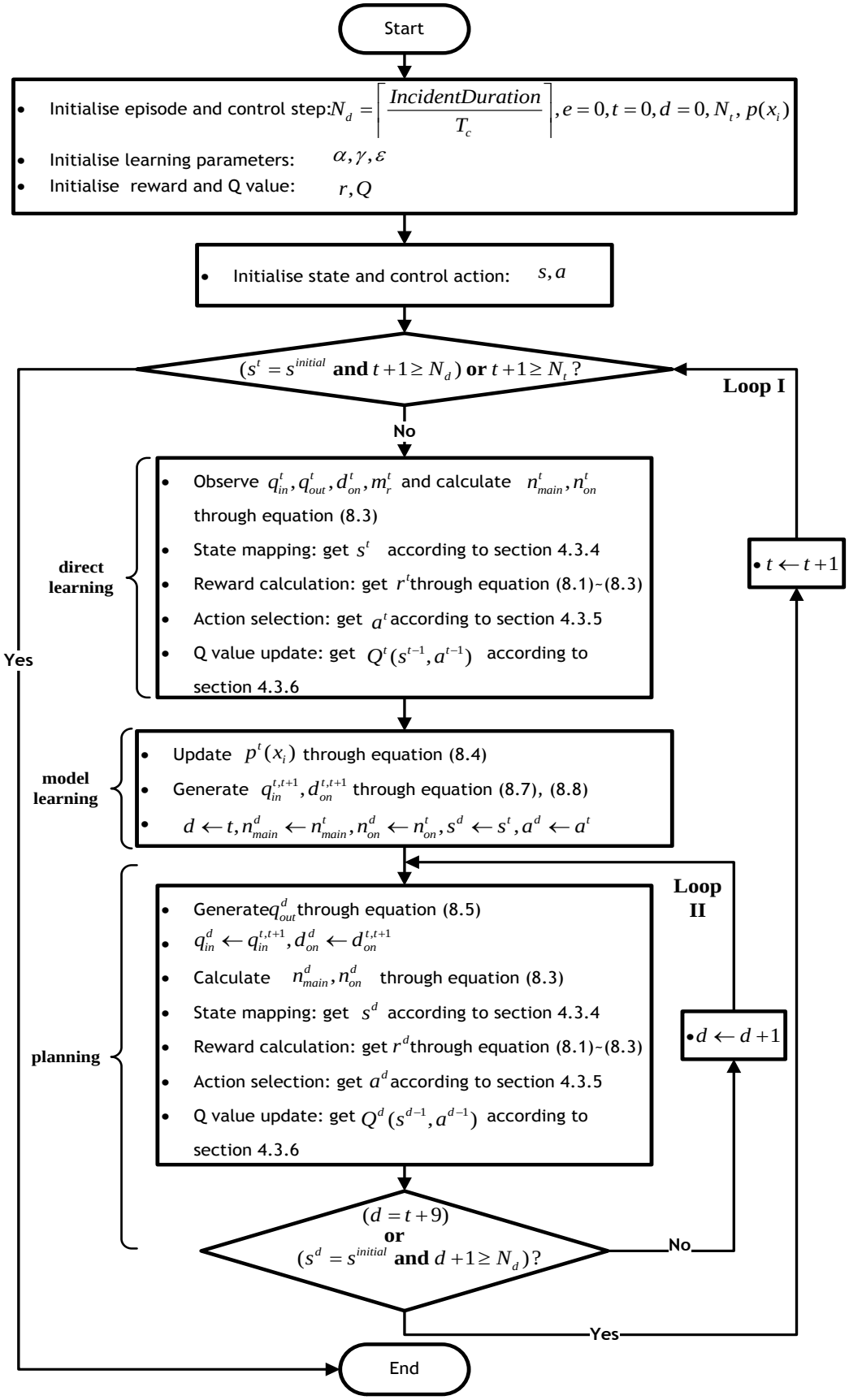


Figure 8.7: IRL algorithm

8.4 Simulation Experiments

Taking the non-controlled (NC) situation as the base line, a series of experiments is designed to compare the proposed IRL algorithm with two other methods, one is the standard RAS and the other is the widely used algorithm ALINEA (Papageorgiou et al. 1991). Experiments and relevant results are described as follows.

8.4.1 Link with AIMSUN

The AIMSUN used in this chapter is version 6.1. RAS can be linked with AIMSUN through the API (application program interface) provided by AIMSUN.

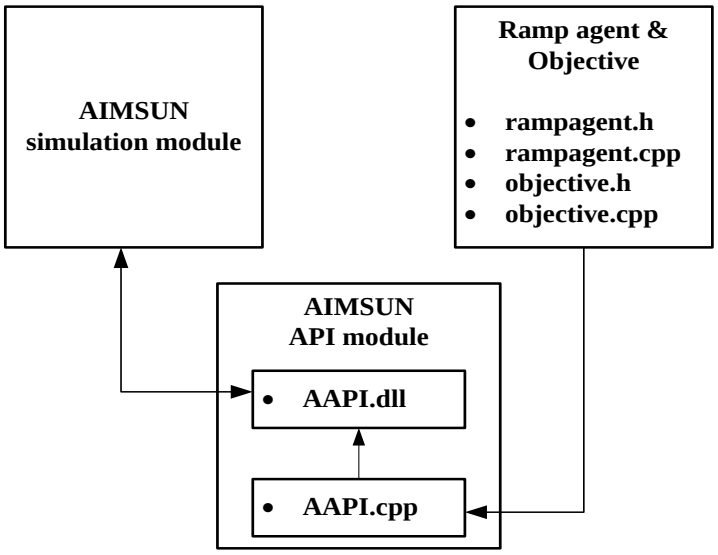


Figure 8.8: Connection between AIMSUN and RAS

Figure 8.8 shows the connection between RAS and AIMSUN. Functions embed in the files “rampagent.cpp” and “objective.cpp” can be linked with AIMSUN API in the file “APPI.cpp” which can be further converted to a DLL (dynamic-link library) file “AAPI.dll”. Through this DLL file, the ramp agent can capture raw traffic information from AIMSUN and generate suitable control actions to control the simulated motorway traffic in AIMSUN. This kind of interaction can be implemented at each control step. One simulation

run of the AIMSUN corresponds to one episode of RAS. The main difference between original RAS and IRL is the function ***startStateTransition()*** (in the file “rampagent.cpp”), by replacing this function with the one introduced in Section 8.3.3, the same way mentioned here can be used to link IRL and AIMSUN.

8.4.2 Network layout

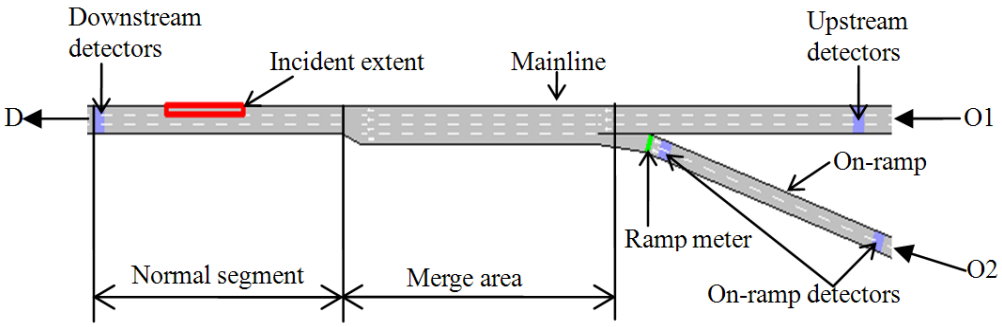


Figure 8.9: Layout of the analysed motorway segment

A simple network with two origins and one destination is used for the analysis. This network contains a typical motorway segment composed of a three-lane mainline and a two-lane on-ramp (Figure 8.9). O1 and O2 represent the origins of mainline and on-ramp traffic, respectively. D is the shared destination of trips from both O1 and O2. Detectors on the mainline and on-ramp are used to capture the real-time traffic arrival and departure rates of the motorway segment.

In this study, an incident with only one lane blocked is considered. The incident is located in the outer lane of the normal segment as shown in Figure 8.9. Detector spacing on the mainline and on-ramp is 1000 m and 350 m, respectively. The merge area and normal segment have the same length, 250 m. The incident extent is 80 m that is assumed to be constant during the incident.

8.4.3 Scenarios and parameters

The simulation experiment is designed to cover one and a half hours from 8:00 to 9:30. After 30 minutes of normal operation (for warm-up), the incident is triggered at 8:30 and lasts for 30 minutes. Three scenarios with different traffic demand (high, medium and low) of O1 and O2 are considered as follows:

(1) Scenario A: O1: 3000 vehs/h, O2: 900 vehs/h

(2) Scenario B: O1: 2700 vehs/h, O2: 750 vehs/h

(3) Scenario C: O1: 2400 vehs/h, O2: 600 vehs/h

For ALINEA, the default parameters provided by AIMSUN are used: the regulator parameter is 70 veh (this is a commonly used parameter for ALINEA and has shown its effectiveness in many field tests (Papageorgiou et al. 1997)), the desired downstream occupancy is 26 percent, and the calculation interval is the same as the control interval (1 minute). For IRL and RAS, learning parameters are the same (α=0.2, γ=0.75, ε=0.01). Other parameters related to the traffic network can be set according to the limitations of motorway storage space and historical traffic data (collected through running AIMSUN without control), which are summarised in Table 8.I. In this study, three weight values (0.8, 0.6 and 0.5) denoting different importance assignment for the mainline and on-ramp traffic are considered for comparison.

Table 8.1: Parameters for RAS and IRL

Parameter	q_{cap}	$\hat{n}_{main}^{max}$	$\hat{n}_{main}^{min}$	$\hat{n}_{on}^{max}$	$\hat{n}_{on}^{min}$	n_{main}^{max}	n_{on}^{max}
Value	111.5	26	-28	13	-8	650	140

8.4.4 Results

The IRL algorithm in this study is aiming to save the travel time on the motorway mainline (not the whole network) without making the on-ramp queue exceed its storage space and worsening other environmental problems such as vehicle emissions. In this case, the comparison between IRL, RAS and ALINEA is conducted from three aspects: mainline total travel time (TTT)[12], on-ramp queue length, CO2 emissions of the whole network and the learning speed (this is just related to IRL and RAS). Both RAS and IRL run for 5000 iterations before the experiment.

Table 8.2: Mainline total travel time comparison

Scenario	ξ	Mainline TTT comparison						
		NC (min)	ALINEA		RAS		IRL	
			Value (min)	Dec* (%)	Value (min)	Dec (%)	Value (min)	Dec* (%)
A	0.8	6630.3	6045.1	9.7	5535.6	16.5	3989.6	39.8
	0.6	6630.3	6045.1	9.7	5543.3	16.4	4955.0	25.3
	0.5	6630.3	6045.1	9.7	5573.4	15.9	5023.5	24.2
B	0.8	3925.9	3701.9	6.1	3066.0	21.9	2724.2	30.6
	0.6	3925.9	3701.9	6.1	3132.2	20.2	2808.9	28.5
	0.5	3925.9	3701.9	6.1	3337.7	15.0	3054.7	22.2
C	0.8	2287.9	2425.6	-5.7	2385.5	-4.3	2282.8	0.2
	0.6	2287.9	2425.6	-5.7	2406.2	-5.2	2320.0	-1.4
	0.5	2287.9	2425.6	-5.7	2462.8	-7.6	2326.2	-1.7

* Dec means decrease compared with NC

[12] The mainline total travel time (TTT) used by AIMSUN is not the same as the TTT in a macroscopic model mentioned in chapter 4. In AIMSUN, TTT is defined at the micro level, which is the sum of travel times experienced by all the vehicles that have crossed the mainline.

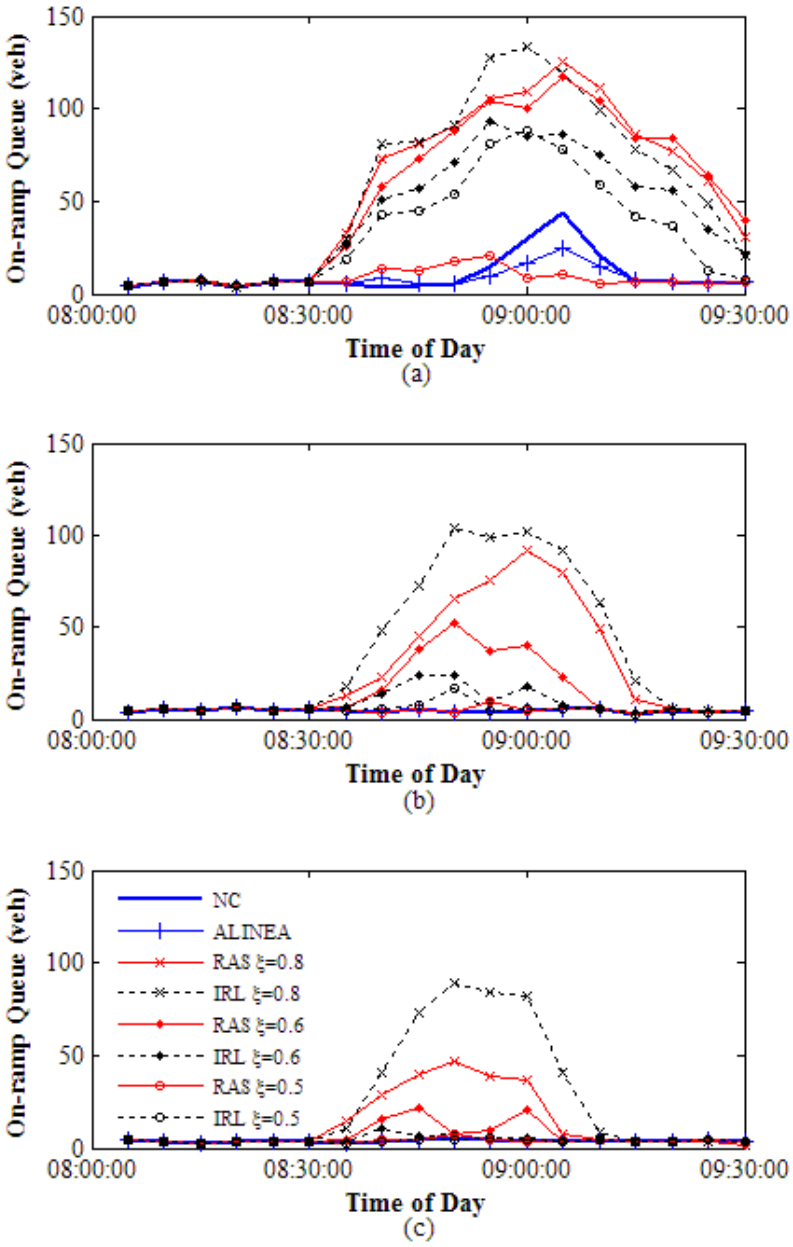


Figure 8.10: On-ramp queue length comparison for (a) scenario A, (b) scenario B, (c) scenario C

(1) A comparison of the mainline total travel time (TTT) can be found in Table 8.2. In most situations, RAS and IRL can outperform ALINEA on the TTT saving in the mainline. This is mainly because ALINEA cannot regulate its target according to changed road capacity. ALINEA always tries to obtain the predefined downstream occupancy, which will lead to severe congestion in the mainline. Compared with RAS, IRL can always

produce a better performance especially in the high and medium demand situations (A, B). This is because IRL learns faster than RAS. After the same number of iterations (5000), IRL can learn a better control strategy than RAS, which is closer to the optimal solution. Under the low demand scenario C, congestion does not always exist. In this situation, all three strategies cannot reduce the mainline TTT (although IRL can reduce TTT by 0.2%, this reduction is too small). This result is reasonable, because the traffic efficiency cannot be improved when congestion does not occur (Zhang et al. 1996). In this situation, the motorway should be left without control.

(2) A comparison of on-ramp queues can be found in Figure 8.10. The queue length in the simulation experiments is captured every 5 minutes. RAS and IRL can keep the queue length under the maximum allowed value (140 vehicles limited by the storage space) in all scenarios. The weight value has great impacts on the on-ramp queue. A high weight value means that more importance is assigned to the mainline traffic. This leads to longer on-ramp queue, especially in the high demand scenario A. ALINEA keeps the queue length at a low level close to the non-controlled situation in all scenarios.

(3) A comparison of total CO2 emissions for the whole network during the incident is shown in Figure 8.11. CO2 emissions can be estimated using the model proposed by (Int Panis et al. 2006), which has been imbedded in AIMSUN. Compared with NC, all three algorithms will increase CO2 emissions in the low demand scenario C. In scenario B, RAS with weight value 0.5, IRL with weight values 0.6 and 0.5 outperform ALINEA. However, in scenario A, except for RAS with a weight of 0.5, the RAS and IRL algorithms cannot work as well as ALINEA.

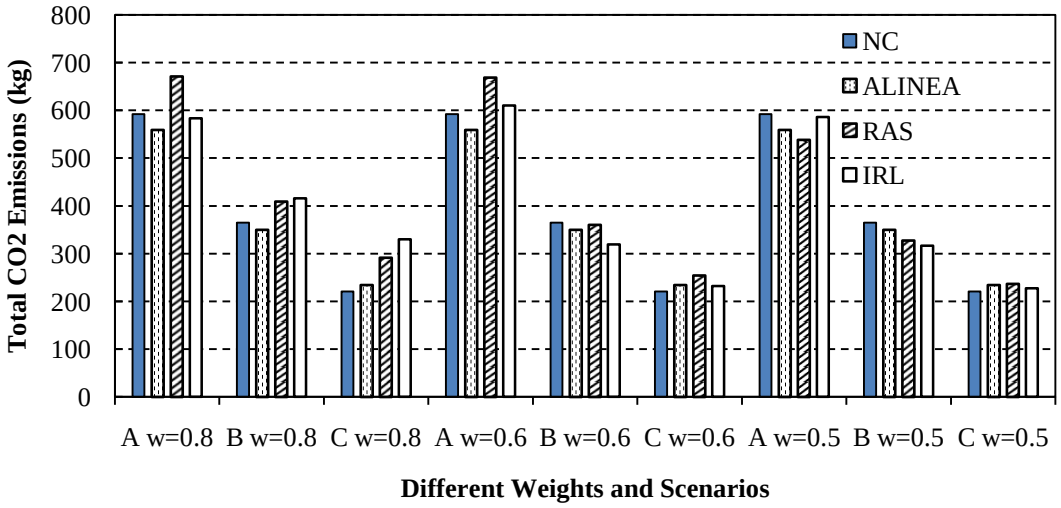


Figure 8.11: Total CO_2 emissions during the incident

8.5 Summary and Conclusions

The aim of this chapter is to test RAS and its extension IRL in an uncertain incident situation. The reusability of RAS has been shown by successfully linking the RAS-related files with AIMSUN in a number of simulation experiments. Through comparative experiments, it is found that: (1) both IRL and RAS can outperform ALINEA on reducing the mainline TTT during the incident, (2) with the same number of iterations (5000), IRL can outperform RAS in almost all scenarios, which means IRL can learn faster than RAS, (3) In the medium demand scenario, IRL can reduce both mainline TTT and vehicle emissions with suitable weights (such as 0.6 and 0.5).

Although some positive results of IRL have been shown in this chapter, as a preliminary test, many situations such as different motorway networks (e.g. with off-ramps or multiple on-ramps), incident locations (e.g. in the merge area or near the on-ramp), incident extents (e.g. two lanes are blocked) and other weight values have not been involved. As incident-induced congestion is considered as an extension to the main work presented in this thesis, all the aforementioned situations lie outside the scope of this study, but can be left for future work.

CHAPTER 9 CONCLUSIONS AND FUTURE WORK

This chapter summarises the main achievements in the thesis and gives possible directions for future work. The organisation of this chapter is as follows. Firstly, Section 9.1 summarises how the research objectives of this doctoral research are achieved. Then, the main contributions of this thesis to the RL-based ramp metering control are presented in Section 9.2. Some limitations of the current work and several possible directions for future work are discussed in Section 9.3. Finally, Section 9.4 gives a final remark of the whole thesis.

9.1 Research Summary

A self-learning motorway traffic control system has been developed in this doctoral research to deal with ramp metering problems. This system is named RAS (ramp agent system) which contains a group of ramp agents that can work independently or cooperatively with shared information to achieve predefined control objectives. To overcome some limitations of previous RL-based systems, five research objectives were proposed in Chapter 1. In this section, how these objectives were achieved through this study is summarised. Firstly, a brief review of research problems, their related research objectives and where each research objective has been achieved is shown Table 9.1. After that, a detailed summary about how these research objectives were attained is given in this section.

Table 9.1: Summary of research objectives

Research Problems	Research Objectives	Supporting Chapters and Sections
	(1) To investigate the state of the art of RL technology and its applications in the ramp metering domain, including both local and coordinated RL-based systems.	Chapters 2 and 3
(1) There is a lack of a general framework for designing a RL-based system for ramp metering application, and each study has its own way to define RL elements.	(2) To provide a general framework for designing a RL-based ramp metering system, regarding the definitions of RL elements, the structure and modules of a RL-based system.	Chapter 4 (Sections 4.1, 4.2 and 4.3)
(2) Although a few studies have considered the coordination problems in a RL-based strategy, improving motorway efficiency is still the main concern. How to add new objectives such as user equity and balance different control objectives have not been well studied.	(3) To explore the application of RL to ramp metering for both single- and multi-objective problems under the framework proposed in Objective (2). Two different control objectives with two different control algorithms are developed and analysed.	Chapter 4 (Sections 4.3 and 4.4)
(3) There is a lack of systematic evaluation for a RL-based system regarding the influence of learning parameters and the effectiveness of algorithms on different networks.	(4) To develop a platform with initial software implementations based on Objectives (2) and (3), which can be used to evaluate the RL-based system	Chapter 5
	(5) To evaluate the proposed system based on Objective (4) by conducting simulation-based experiments considering both hypothetical and real traffic networks.	Chapters 6, 7 and 8

Objective (1): ***To investigate the state of the art of RL technology and its applications in the ramp metering domain***

This objective is the basis for all other four research objectives. Through the review of RL-based ramp metering systems in Chapter 2, three problems (or limitations) regarding the system design and evaluation were found (as shown in Table 9.1), based on which four specific research objectives were proposed for this study. Both theoretical and technical issues of RL were investigated in Chapter 3, through which the main mechanism and algorithms of RL can be obtained. Specifically, two kinds of the basic RL algorithms, namely Q-learning and linear scalarised Q-learning were found to be suitable for building a ramp metering system and used in the following chapter about the system design in this study.

Objective (2): ***To provide a general framework for designing a RL-based ramp metering system***

A general framework for designing a RL-based system was presented in Chapter 4, which contains the definition of three elements (i.e. state, action and reward), the structure of the ramp agent with related modules that can accomplish the learning process.

- The structure and modules relating to a ramp agent were described in Sections 4.1 and 4.3. This structure contains two main modules, namely the ***RampAgent*** module and the ***Objective*** module. The advantage of a separate ***Objective*** module is that it contains general functions for reward calculation, Q value updating and scalarisation. Based on these functions, different rewards according to different control objectives can be defined under the same general framework.

- The elements of a ramp agent were defined in Section 4.3. The Markov property is considered in the definition of three elements, especially the

state and action. In this way, the state transition of a ramp agent satisfies the Markov decision process, which can guarantee the effectiveness of RL on ramp metering problems.

Objective (3): ***To explore the application of RL to ramp metering for both single- and multi-objective problems***

To deal with both single and multiple control objectives, in Chapter 4, RAS was designed to contain two working modes, i.e. single-objective mode and multi-objective mode with two corresponding control algorithms. In this study, two control objectives, namely improving traffic efficiency by reducing TTS (total time spent on motorways) and maintaining user equity by balancing TWT (total waiting time at on-ramps) were considered. According to these two objectives, two rewards were derived from a common definition of TTS and TWT. Given different control objectives and their corresponding rewards, control algorithms (for single- and multi-objective modes) can integrate relevant functions from different modules (developed in Objective (2)) and solve ramp metering problems by generating optimal metering rates at each time step.

- A single-objective algorithm was developed in Section 4.4.1 for the single-objective mode. Only the reward related to TTS was involved in this control algorithm. Under the single-objective mode, ramp agents of RAS can work independently to achieve the relevant objective (reducing TTS) based on the Q-learning mechanism.
- A multi-objective algorithm was developed in Section 4.4.2 for the multi-objective mode. Both rewards regarding TTS and TWT were considered in this algorithm. Through sharing information about the on-ramp queues, ramp agents in RAS can balance TWT at different on-ramps while trying

to reduce TTS. The multi-objective algorithm was derived from the linear scalarised Q-learning algorithm.

Objective (4): *To develop a platform with initial software implementations*

A software platform containing reusable classes for RAS and ACTM (asymmetric cell transmission model) was developed in Chapter 5. All classes relating to RAS and ACTM were programmed by C++, which can be used to evaluate the proposed system.

- Although the main purpose of ACTM in this study is to evaluate RAS, it can also be used to test other control strategies such as ALINEA mentioned in Chapter 6.
- The implementation of RAS is completely independent from ACTM. It can be reused by other traffic flow models under the same programming framework (using C++). For instance, in Chapter 8, the reusability of RAS was tested with the microscopic simulation software AIMSUN.

Objective (5): *To evaluate the proposed system using simulation-based experiments*

To evaluate the proposed RAS, a series of simulation-based experiments from three case studies, namely single-ramp, multi-ramp and real-network cases were designed and conducted in Chapters 6 and 7. The performance of RAS in three cases was compared with one of the most widely used ramp metering strategies, ALINEA.

- In the single-ramp case (Section 6.1), only one ramp agent with one controlled on-ramp was considered. One important issue considered in this case is how to select suitable learning parameters for a ramp agent. Through the sensitivity analysis about three parameters α (learning

rate), γ (discount rate) and ε (action selection parameter), it was found that the parameters with α = 0.2, γ = 0.75 and ε = 0.01 could achieve a good performance which were used for the following tests of Chapters 6 and 7. With these selected parameters, both the abilities of RAS to improve traffic efficiency and manage the on-ramp queue under predefined constraints were tested. As shown in Section 6.1.5, by setting suitable constraints for the normalised reward, RAS can successfully restrict the on-ramp queue under the permitted length. However, for the efficiency test in Section 6.1.4, RAS showed no advantages to improve efficiency compared with ALINEA. Both RAS and ALINEA reduced the network TTS to the same level (around 3920 veh.min).

- In the multi-ramp case (Section 6.2), the performance of RAS composed of multiple ramp agents was tested. In this case, both single- and multi-objective algorithms were analysed. As shown in Section 6.2.2 to 6.2.4, with the single-objective algorithm, RAS outperformed ALINEA on improving efficiency when multiple on-ramps could cause congestion, while it showed no improvement when only one on-ramp could cause congestion. Besides efficiency improvement, an additional objective about user equity (balancing TWT) was involved in the multi-objective algorithm shown in Scetion 6.2.5. With necessary information sharing about the on-ramp queues, two control objectives can be achieved together with a suitable compromise by properly setting the weights of the multi-objective algorithm. RAS provided a huge advantage over ALINEA in terms of maintaining equity. As shown in Section 6.2.5, RAS-EQ could keep the stand deviation of TWT (SD(TWT)) close to 0 veh.min, while ALINEA corresponded to a SD(TWT) higher than 200 veh.min.

- In the real-network case (Chapter 7), a stretch of the M6 motorway in the UK was selected for the case study. The real network layout and traffic data in morning peak hours were considered in this case. Fluctuating demand flows and split ratios were extracted from the highways agency traffic information system (HATRIS), based on which both single- and multi-objective modes of RAS were tested. Through the simulation experiments, RAS showed its effectiveness in dealing with fluctuating real traffic conditions. Similar to the multi-ramp case, RAS was much better than ALINEA in terms of maintaining user equity. As shown in Section 7.4.4, RAS-EQ provided a much lower SD(TWT) (5.7 veh.min) than ALINEA (2105.8 veh.min).

Besides the evaluation of RAS in the recurrent congestion situation presented in Chapters 6 and 7, an extension of RAS was tested in Chapter 8 where the non-recurrent congestion caused by incidents was used as a test. The extension of RAS (named IRL) was based on the Dyna-Q architecture, and compared with ALINEA and RAS (RAS without extension). With suitable parameter settings, IRL outperformed both ALINEA and RAS on reducing mainline total travel time without increasing the vehicle emissions.

9.2 Research Contributions

Through achieving five research objectives, this study has made its contributions to the applications of RL in the ramp metering domain. Some key contributions are highlighted as follows:

- Two rewards were defined for two different control objectives, i.e. traffic efficiency and user equity. These two rewards were derived from the commonly used definitions of total time spent on the motorway (efficiency-related) and total waiting time at on-ramps (equity-related). So far, the equity-related objective has not been well studied in the RL-

based systems, and there has been very little research into the definition of equity-related reward. To the best knowledge of the author of this thesis, the only attempt can be found so far is shown in (Zhaohui and Kaige 2010). Although the equity issue was considered in this work, how to derive an equity-related reward was not clearly described. In the research presented in this thesis, the equity-related reward was defined to balance the total waiting times at different on-ramps in real time, and the effectiveness of this reward has been proved in the following tests.

- A general definition of state and action that satisfies the Markov property was proposed, which provided a clear way to define RL-related elements for ramp metering problems. RL is a learning method that can solve the MDP problems without models. Thus, an effective RL problem should be an MDP problem and satisfy the Markov property. However, this issue was neglected by most of existing studies. Each study used its own way to define state and action without enough explanations for why their definitions worked. In this study, the definition of state and action was derived from the vehicle conservation (a commonly accepted conservation law in traffic flow models) and satisfied the Markov property, which guaranteed the effectiveness of this definition.

- Two control algorithms for both single- and multi-objective problems were developed. These two algorithms were derived from the basic Q-learning and linear scalarised Q-learning algorithms, which can be used to deal with single- and multi-objective problems, respectively. Most of existing RL-related studies focused on using Q-learning to solve single-objective problems in the ramp metering domain. When multiple control objectives are involved, the basic Q-learning will become invalidated. This thesis provided a possible solution to solve multi-objective ramp metering problems using the linear scalarised Q-learning.

- A software platform was developed in this study. The reusability of this platform provided a flexible way to test RL-based algorithms and other ramp metering strategies using simulation-based experiments. This platform was programmed by C++ containing the implementations of RAS and ACTM. It has been shown in the thesis that, both of these two parts can be reused, e.g. RAS can be linked with AIMSUN and ACTM can be used to test ALINEA. This platform can be easily extended for use by other researchers in the RL or ramp metering domain.
- A systematic evaluation of RAS was presented in this research. Various aspects of RAS, such as the ability to improve traffic efficiency, manage on-ramp queues and maintain user equity was tested, which proved the effectiveness of RAS in dealing with ramp metering problems. In addition, the competitive relationship between efficiency and equity found in some literature (Kotsialos and Papageorgiou 2004a, Meng and Khoo 2010, Zhang and Levinson 2005) was reaffirmed in the test of maintaining equity in this thesis.

9.3 Research Limitations and Future Work

Although the proposed RAS has shown its effectiveness in many cases, some limitations regarding this new system and related simulation experiments still exist. To overcome these limitations, some possible solutions can be used to form the basis for future work.

- One main limitation of RAS is that it can only tackle a problem with a discrete state set. The main issue faced by the discrete state set is the curse of dimensionality, which means that the Q-table size of a ramp agent will increase exponentially with the growing number of states (Barto and Mahadevan 2003). It will not only increase the burden on the memory of the computer, but also greatly affect the learning time. In this

study, to reduce the memory burden and obtain an acceptable learning time (usually within one hour), the state number of one ramp agent was limited to around 60000. This however led to another problem introduced in Chapter 6. With a limited number of states, the ramp agent was not able to distinguish some critical states with slight differences (such as states around the critical density when severe congestion occurs). In this situation, the ramp agent could fail to find suitable control actions for some states. Thus, as mentioned in Sections 6.2.3 and 6.2.4, the ramp agent could generate unstable control actions (causing unstable density evolution) at locations with severe congestion. One possible solution to this problem is to use the function approximation method. This method can help the ramp agent construct a function that can directly generate Q values from the continuous state values without a Q table (Sutton and Barto 1998). Thus, all state values, not a limited number of discrete states, can be captured and used by the ramp agent. Combining function approximation methods and RAS will be a direction of future work.

- The simulation-based experiments using ACTM only took account of deterministic flow rates (perfect data without noise) from the motorway mainline and on-ramp. Although the fluctuating demands were considered in the real-network case, these demand flows were still kept deterministic for each 15-minute interval. In the practical application, the real-time traffic data are collected with very short time intervals (usually within 1 minute) and has stochastic noise (Kotsialos et al. 2006, Smaragdis and Papageorgiou 2003). This study focused on an ideal traffic environment without the impact of noise. In the future work, a sensitivity analysis can be done to analyse the effects of different types of noise on the performance of RAS (such as its learning speed and

stability), which may provide some advice for field applications of RAS with real-time traffic data.

- Because of the resource limitation, RAS was only compared with one widely used control strategy, ALINEA. Other more advanced control strategies such as model predictive control methods are not considered in this study. It is mainly because these strategies are very difficult to implement and usually need the support of sophisticated algorithms (Hegyi et al. 2005). The codes and related tools for these algorithms are not available at the current stage. If these resources are available, it is worthwhile to compare RAS with these advanced control strategies in future work.

- The field application is the ultimate goal of any control systems. However, the RL-based traffic control systems are still in their early stages and cannot be used to learn from the real-time traffic directly so far (it will take too long in the real-world traffic situation). One solution proposed in (El-Tantawy et al. 2013, Jacob and Abdulhai 2010) suggested that the RL system can learn the optimal control actions from a simulation model first (the simulation can run much faster than the real traffic), and then these actions can be used to control the real traffic. However, the real traffic is highly dynamic which may change after the calibration of simulation models. The control actions obtained from an inaccurate model may not be suitable for the real traffic. Under such circumstances, an alternative way for the field application can be based on the IRL algorithm discussed in Chapter 8. This method can learn from a model and the real traffic simultaneously. The real-time traffic data can be used to improve the model in an on-line situation. This feature has been showed in a preliminary study on IRL presented in this thesis. Further

studies can be done to investigate the stability and adaptability of IRL for different types of models.

- Ramp metering was the only control measure considered in this study. One shortcoming of ramp metering is that it can only control the traffic from on-ramps. For a more effective motorway control, an integrated control system that can control the traffic on both motorway on-ramps and mainline with multiple control measures, such as ramp metering (for on-ramp traffic) and variable speed limit (VSL) (for mainline traffic), is required (Carlson et al. 2014). The ramp agent developed in this study can also be extended to deal with VSL problems by defining three RL elements (state, action and reward) in a VSL scenario.

9.4 Final remark

A self-learning motorway traffic control system focusing on ramp metering has been developed in this thesis. The new system was based on RL and developed following a general framework which overcomes some limitations of previous RL-based applications. The performance of this system was tested through a number of simulation experiments where the effectiveness of the new system in both hypothetical and real networks was shown. Although the RL-based ramp metering systems are still in their early stages, the potential of RL to deal with various tasks, such as improving traffic efficiency, managing queue length and maintaining user equity, has been shown in this thesis, which provides some evidence for improving RL-based systems in future work.

LIST OF REFERENCES

Abdel-Aty, M., Dhindsa, A. and Gayah, V. (2007) Considering various ALINEA ramp metering strategies for crash risk mitigation on freeways under congested regime. *Transportation Research Part C: Emerging Technologies, 15*(2), pp. 113-134.

Arnold Jr, E. (1998) *Ramp metering: a review of the literature*. Charlottesville: Virginia Transportation Research Council.

Bai, X., Zhao, D. and Yi, J. (2009) ADHDP (λ) strategies based coordinated ramps metering with queuing consideration. *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, March 2009, Nashville, USA*. pp. 22-27.

Banks, J. (2000) Are Minimization of delay and minimization of freeway congestion compatible ramp metering objectives? *Transportation Research Record: Journal of the Transportation Research Board, 1727*, pp. 112-119.

Barrett, L. and Narayanan, S. (2008) Learning all optimal policies with multiple criteria. *25th International Conference on Machine learning*, *July 2008, Helsinki, Finland.* pp. 41-47.

Barto, A. G. and Mahadevan, S. (2003) Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems, 13*(4), pp. 341-379.

Becerril-Arreola, R. and Aghdam, A. G. (2007) Decentralised nonlinear control with disturbance rejection for on-ramp metering in highways. *Control Theory & Applications, IET, 1*(1), pp. 253-262.

Benmohamed, L. and Meerkov, S. M. (1994) Feedback control of highway congestion by a fair on-ramp metering. *33rd IEEE Conference on Decision and Control, December 1994, Lake Buena Vista, USA*. pp. 2437-2442.

Bergsma, H. A. (2006) *European ramp metering project (Deliverable D3.2) Simulation testing for local ramp metering strategies.* Brussels: European Commission.

Bhouri, N., Haj-Salem, H. and Kauppila, J. (2013) Isolated versus coordinated ramp metering: Field evaluation results of travel time reliability and traffic impact. *Transportation Research Part C: Emerging Technologies, 28*, pp. 155-167.

Carlson, R. C., Papamichail, I. and Papageorgiou, M. (2014) Integrated feedback ramp metering and mainstream traffic flow control on

motorways using variable speed limits. *Transportation Research Part C: Emerging Technologies, 46*, pp. 209-221.

Cassidy, M. J. and Bertini, R. L. (1999) Some traffic features at freeway bottlenecks. *Transportation Research Part B: Methodological, 33*(1), pp. 25-42.

Chen, C.I., Cruz Jr, J. and Paquet, J. (1974) Entrance ramp control for travel-rate maximization in expressways. *Transportation Research, 8*(6), pp. 503-508.

Chow, A. H. F. and Li, Y. (2014) Robust optimization of dynamic motorway traffic via ramp metering. *IEEE Transactions on Intelligent Transportation Systems, 15*(3), pp. 1374-1380.

Csikós, A. and Varga, I. (2012) Multicriteria ramp metering control considering environmental and traffic aspects. *15th IFAC Workshop on Control Applications of Optimization, September 2012,* Rimini, Italy. pp. 194-199.

Daganzo, C. F. (1994) The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological, 28*(4), pp. 269-287.

Daganzo, C. F. (1995) The cell transmission model, part II: network traffic. *Transportation Research Part B: Methodological, 29*(2), pp. 79-93.

Davarynejad, M., Hegyi, A., Vrancken, J. and van den Berg, J. (2011) Motorway ramp-metering control with queuing consideration using Q-learning. *14th International IEEE Conference on Intelligent Transportation Systems, October 2011, Washington, DC, USA*. pp. 1652-1658.

Dervisoglu, G., Gomes, G., Kwon, J., Horowitz, R. and Varaiya, P. P. (2009) Automatic Calibration of the Fundamental Diagram and Empirical Observations on Capacity. *Transportation Research Board 88th Annual Meeting, January 2009, Washington, DC, USA*.

Dodgson, J., Young, J. and der Veer, J. (2002) *Paying for road use: technical report.* London: National Economic Research Associates.

Dowling, R., Skabardonis, A., Carroll, M. and Wang, Z. (2004) Methodology for Measuring Recurrent and Nonrecurrent Traffic Congestion. *Transportation Research Record: Journal of the Transportation Research Board, 1867*, pp. 60-68.

Dusparic, I. and Cahill, V. (2009) Distributed w-learning: multi-policy optimization in self-organizing systems. *3rd IEEE International Conference on Self-Adaptive and Self-Organizing Systems, September 2009, San Francisco, USA.* pp. 20-29.

El-Tantawy, S., Abdulhai, B. and Abdelgawad, H. (2013) Multiagent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC): Methodology and Large-Scale

Application on Downtown Toronto. *IEEE Transactions on Intelligent Transportation Systems, 14*(3), pp. 1140-1150.

Even-Dar, E. and Mansour, Y. (2004) Learning rates for Q-learning. *The Journal of Machine Learning Research, 5*, pp. 1-25.

Fares, A. and Gomaa, W. (2015) Multi-Agent Reinforcement Learning Control for Ramp Metering. *23rd International Conference on Systems Engineering, August 2014*. pp. 167-173.

Farradyne, P. (2000) *Traffic Incident Management Handbook*. Washington, DC: U.S. Department for Transport.

Fu, L. and Rilett, L. (1997) Real-time estimation of incident delay in dynamic and stochastic networks. *Transportation Research Record: Journal of the Transportation Research Board, 1603*, pp. 99-105.

Gábor, Z., Kalmár, Z. and Szepesvári, C. (1998) Multi-criteria Reinforcement Learning. *15th International Conference on Machine Learning, 1998, Madison, USA*. pp. 197-205.

Gomes, G. and Horowitz, R. (2003) A study of two onramp metering schemes for congested freeways. *the 2003 American Control Conference, June 2003, Denver, USA.* pp. 3756-3761.

Gomes, G. and Horowitz, R. (2006) Optimal freeway ramp metering using the asymmetric cell transmission model. *Transportation Research Part C: Emerging Technologies, 14*(4), pp. 244-262.

Grant-Muller, S. and Larid, J. (2007), The cost of congestion. *European Transport Conference 2007, October 2007, Noordwijkerhout, Netherlands*.

Greenlee, T. L. and Payne, H. J. (1977) Freeway ramp metering strategies for responding to incidents. *1977 IEEE Conference on Decision and Control including the 16th Symposium on Adaptive Processes and A Special Symposium on Fuzzy Set Theory and Applications, December 1977, New Orleans, USA*. pp. 987-992.

Greenshields, B. D., Bibbins, J., Channing, W. and Miller, H. (1935) A study of traffic capacity. *14th Annual Meeting of Highway Research Board, December 1934, Washington, DC, USA.* pp. 448-477.

Haddad, J., Ramezani, M. and Geroliminis, N. (2013) Cooperative traffic control of a mixed network with two urban regions and a freeway. *Transportation Research Part B: Methodological, 54*, pp. 17-36.

Hadi, M., Sinha, P. and Wang, A. (2007) Modeling reductions in freeway capacity due to incidents in microscopic simulation models. *Transportation Research Record: Journal of the Transportation Research Board, 1999*, pp. 62-68.

Hall, F. L. and Agyemang-Duah, K. (1991) Freeway capacity drop and the definition of capacity. *Transportation Research Record, 1320,* pp. 91-98.

Hall, R. W. (1993) Non-recurrent congestion: How big is the problem? Are traveler information systems the solution? *Transportation Research Part C: Emerging Technologies, 1*(1), pp. 89-103.

Han, X. and Naeher, L. P. (2006) A review of traffic-related air pollution exposure assessment studies in the developing world. *Environment international, 32*(1), pp. 106-120.

Hegyi, A., De Schutter, B. and Hellendoorn, H. (2005) Model predictive control for optimal coordination of ramp metering and variable speed limits. *Transportation Research Part C: Emerging Technologies, 13*(3), pp. 185-209.

Highways agency. (2007) *Ramp metering summary report*. Bristol: Highways Agency Publications.

Hou, Z., Xu, J.-X. and Yan, J. (2008) An iterative learning approach for density control of freeway traffic flow via ramp metering. *Transportation Research Part C: Emerging Technologies, 16*(1), pp. 71-97.

Humphrys, M. (1995) *W-learning: Competition among selfish Q-learners*. Cambridge: Computer Laboratory, University of Cambridge.

Int Panis, L., Broekx, S. and Liu, R. (2006) Modelling instantaneous traffic emission and the influence of traffic speed limits. *Science of the Total Environment, 371*(1), pp. 270-285.

Jacob, C. and Abdulhai, B. (2006) Automated adaptive traffic corridor control using reinforcement learning: approach and case studies. *Transportation Research Record: Journal of the Transportation Research Board,* (1959), pp. pp 1-8.

Jacob, C. and Abdulhai, B. (2010) Machine learning for multi-jurisdictional optimal traffic corridor control. *Transportation Research Part A: Policy and Practice, 44*(2), pp. 53-64.

Jacobson, L. N., Henry, K. C. and Mehyar, O. (1989) Real-time metering algorithm for centralized control. *Transportation Research Record: Journal of the Transportation Research Board, 1232*, pp. 17-26.

Jiang, R. and Chung, E. (2013) Modified cell transmission model for better ramp metering evaluation. *OPTIMUM 2013-International Symposium on Recent Advance in Transport Modelling, April 2013, Kingscliff, Australia*.

Jiuh-Biing, S. and Mei-Shiang, C. (2007) Stochastic optimal-control approach to automatic incident-responsive coordinated ramp control. *IEEE Transactions on Intelligent Transportation Systems, 8*(2), pp. 359-367.

Kaelbling, L., Littman, M. and Moore, A. (1996) Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research, 4*, pp. 237-285.

Kim, Y. and Hall, F. (2004) Relationships Between occupancy and density reflecting average vehicle lengths. *Transportation Research Record: Journal of the Transportation Research Board, 1883*, pp. 85-93.

Kotsialos, A. and Papageorgiou, M. (2001) Efficiency versus fairness in network-wide ramp metering. *4th IEEE Conference on Intelligent Transportation Systems, August 2001, Oakland, USA.* pp. 1189-1194.

Kotsialos, A. and Papageorgiou, M. (2004a) Efficiency and equity properties of freeway network-wide ramp metering with AMOC. *Transportation Research Part C: Emerging Technologies, 12*(6), pp. 401-420.

Kotsialos, A. and Papageorgiou, M. (2004b) Motorway network traffic control systems. *European Journal of Operational Research, 152*(2), pp. 321-333.

Kotsialos, A., Papageorgiou, M., Hayden, J., Higginson, R., McCabe, K. and Rayman, N. (2006) Discrete release rate impact on ramp metering performance. *Intelligent Transport Systems, IEE Proceedings, 153*(1), pp. 85-96.

Kotsialos, A., Papageorgiou, M. and Middelham, F. (2001) Optimal Coordinated Ramp Metering with Advanced Motorway Optimal Control. *Transportation Research Record: Journal of the Transportation Research Board, 1748*, pp. 55-65.

Lau, R. (1997) *Ramp metering by zone—The Minnesota algorithm*. Minnesota: Minnesota Department of Transportation.

Lee, C., Hellinga, B. and Ozbay, K. (2006) Quantifying effects of ramp metering on freeway safety. *Accident Analysis & Prevention, 38*(2), pp. 279-288.

Levinson, D. and Zhang, L. (2006) Ramp meters on trial: Evidence from the Twin Cities metering holiday. *Transportation Research Part A: Policy and Practice, 40*(10), pp. 810-828.

Levinson, D., Zhang, L., Das, S. and Sheikh, A. (2002) Evaluating ramp meters: evidence from the Twin Cities ramp meter shut-off. *Transportation Research Board 81st Annual Meeting, January 2002, Washington, DC, USA*.

Lewis, F. L. and Vrabie, D. (2009) Reinforcement learning and adaptive dynamic programming for feedback control. *Circuits and Systems Magazine, IEEE, 9*(3), pp. 32-50.

Li, J., Lan, C. J. and Gu, X. (2006) Estimation of incident delay and its uncertainty on freeway networks. *Transportation Research Record: Journal of the Transportation Research Board, 1959*, pp. 37-45.

Lighthill, M. J. and Whitham, G. B. (1955) On kinematic waves. II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences, 229*(1178), pp. 317-345.

Lu, C., Chen, H. and Grant-Muller, S. (2013) An indirect reinforcement learning approach for ramp control under incident-induced congestion. *16th International IEEE Conference on Intelligent Transportation Systems, October 2013, The Hague, Netherlands*. pp. 979-984.

Lu, C., Chen, H. and Grant-Muller, S. (2014) Indirect reinforcement learning for incident-responsive ramp control. *Procedia-Social and Behavioral Sciences, 111,* pp. 1112-1122.

Masher, D. P., Ross, D., Wong, P., Tuan, P., Zeidler, H. and Petracek, S. (1975) *Guidelines for design and operation of ramp control systems*. Stanford: Stanford Research Institute.

May, A. D. (1990) *Traffic flow fundamentals*. New Jersey: Prentice Hall.

Meng, Q. and Khoo, H. L. (2010) A Pareto-optimization approach for a fair ramp metering. *Transportation Research Part C: Emerging Technologies, 18*(4), pp. 489-506.

Muñoz, L., Sun, X., Horowitz, R. and Alvarez, L. (2003) Traffic density estimation with the cell transmission model. *the 2003 American Control Conference, June 2003, Denver, USA*. pp. 3750-3755.

Muñoz, L., Sun, X., Sun, D., Gomes, G. and Horowitz, R. (2004) Methodological calibration of the cell transmission model. *the 2004 IEEE American Control Conference, June 2004, Boston, USA*. pp. 798-803.

Ngatchou, P., Anahita, Z. and El-Sharkawi, M. A. (2005) Pareto Multi Objective Optimization. *13th International Conference on Intelligent Systems Application to Power Systems, November 2005, Arlington, USA*. pp. 84-91.

Noland, R. B. and Quddus, M. A. (2005) Congestion and safety: A spatial analysis of London. *Transport Research Part A: Policy and Practice, 39*(7), pp. 737-754.

Ozbay, K. and Kachroo, P. (1999) *Incident management in intelligent transportation systems,* Boston: Artech House.

Paesani, G., Kerr, J., Perovich, P. and Khosravi, F. (1997) System wide adaptive ramp metering (SWARM). *ITS America 7th Annual Meeting and Exposition: Merging the Transportation and Communications Revolutions, June 1997, Washington, DC, USA*.

Papageorgiou, M. (1980) A new approach to time-of-day control based on a dynamic freeway traffic model. *Transportation Research Part B: Methodological, 14*(4), pp. 349-360.

Papageorgiou, M., Blosseville, J.-M. and Haj-Salem, H. (1990) Modelling and real-time control of traffic flow on the southern part of Boulevard Peripherique in Paris: Part II: Coordinated on-ramp metering. *Transportation Research Part A: General, 24*(5), pp. 361-370.

Papageorgiou, M., Diakaki, C., Kotsialos, A. and Wang (2003) Review of road traffic control strategies. *Proceedings of the IEEE, 91*(12), pp. 2041-2042.

Papageorgiou, M., Hadj-Salem, H. and Blosseville, J.M. (1991) ALINEA: A local feedback control law for on-ramp metering. *Transportation Research Record: Journal of the Transportation Research Board, 1320,* pp. 58-64.

Papageorgiou, M., Hadj-Salem, H. and Middelham, F. (1997) ALINEA local ramp metering: Summary of field results. *Transportation Research Record: Journal of the Transportation Research Board, 1603*(1), pp. 90-98.

Papageorgiou, M., Kosmatopoulos, E., Papamichail, I. and Wang, Y. (2007) ALINEA maximises motorway throughput-an answer to flawed criticism. *Traffic Engineering and Control, 48*(6), pp. 271.

Papageorgiou, M. and Kotsialos, A. (2002) Freeway ramp metering: an overview. *IEEE Transactions on Intelligent Transportation Systems, 3*(4), pp. 271-281.

Papamichail, I., Kotsialos, A., Margonis, I. and Papageorgiou, M. (2010) Coordinated ramp metering for freeway networks – A model-predictive hierarchical control approach. *Transportation Research Part C: Emerging Technologies, 18*(3), pp. 311-331.

Pinnell, C., Drew, D. R., McCasland, W. R. and Wattleworth, J. A. (1967) Evaluation of entrance ramp control on a six-mile freeway section. *Highway Research Board,* (157)*,* pp. 22-76.

Prevedouros, P., Halkias, B., Papandreou, K. and Kopelias, P. (2008) Freeway incidents in the United States, United Kingdom, and Attica Tollway, Greece: characteristics, available capacity, and models. *Transportation Research Record: Journal of the Transportation Research Board, 2047*, pp. 57-65.

Puterman, M. L. (2009) *Markov decision processes: discrete stochastic dynamic programming,* John Wiley & Sons.

Rezaee, K., Abdulhai, B. and Abdelgawad, H. (2012) Application of reinforcement learning with continuous state space to ramp metering in real-world conditions. *15th International IEEE Conference on Intelligent Transportation Systems, September 2012, Anchorage, USA.* pp. 1590-1595.

Rezaee, K., Abdulhai, B. and Abdelgawad, H. (2013) Self-Learning adaptive ramp metering: analysis of design parameters on a test case in Toronto, Canada. *Transportation Research Record: Journal of the Transportation Research Board, 2396*, pp. 10-18.

Rezaee, K., Abdulhai, B. and Abdelgawad, H. (2014) Closed-Loop Optimal Freeway Ramp Metering Using Continuous State Space

Reinforcement Learning with Function Approximation. *Transportation Research Board 93rd Annual Meeting, January 2014, Washington DC*.

Richards, P. I. (1956) Shock waves on the highway. *Operations research, 4*(1), pp. 42-51.

Rodgers, S., Wadsworth, B., Smith, S. D. and Forde, M. C. (2006) An analysis of road traffic incidents on the M25 motorway, UK. *Proceedings of the ICE-Transport, 159*(1), pp. 1-8.

Roijers, D. M., Vamplew, P., Whiteson, S. and Dazeley, R. (2013) A Survey of Multi-Objective Sequential Decision-Making. *Journal of Artificial Intelligence Research, 48*, pp. 67-113.

Saltelli, A. (1999) Sensitivity analysis: Could better methods be used? *Journal of Geophysical Research: Atmospheres, 104*(D3), pp. 3789-3793.

Schwartz, S. C. and Tan, H. H. (1977) Integrated control of freeway entrance ramps by threshold regulation. *1977 IEEE Conference on Decision and Control including the 16th Symposium on Adaptive Processes and A Special Symposium on Fuzzy Set Theory and Applications, December 1977, New Orleans, USA*. pp. 984-986.

Skabardonis, A., Varaiya, P. and Petty, K. (2003) Measuring Recurrent and Nonrecurrent Traffic Congestion. *Transportation Research Record: Journal of the Transportation Research Board, 1856*, pp. 118-124.

Smaragdis, E. and Papageorgiou, M. (2003) Series of New Local Ramp Metering Strategies. *Transportation Research Record: Journal of the Transportation Research Board, 1856*, pp. 74-86.

Smith, B. L., Qin, L. and Venkatanarayana, R. (2003) Characterization of freeway capacity reduction resulting from accidents. *Journal of Transportation Engineering, 129*(4), pp. 362-368.

Sun, X. and Horowitz, R. (2006) Set of New Traffic-Responsive Ramp-Metering Algorithms and Microscopic Simulation Results. *Transportation Research Record: Journal of the Transportation Research Board, 1959*, pp. 9-18.

Sutton, R. S. (1991) Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulentinl, 2*(4), pp. 160-163.

Sutton, R. S. and Barto, A. G. (1998) *Reinforcement learning: an introduction.* London: MIT press.

Tian, Q., Huang, H.J., Yang, H. and Gao, Z. (2012) Efficiency and equity of ramp control and capacity allocation mechanisms in a freeway corridor. *Transportation Research Part C: Emerging Technologies, 20*(1), pp. 126-143.

Valenti, G., Lelli, M. and Cucina, D. (2010) A comparative study of models for the incident duration prediction. *European Transport Research Review, 2*(2), pp. 103-111.

Vamplew, P., Dazeley, R., Berry, A., Issabekov, R. and Dekker, E. (2011) Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning, 84*(1-2), pp. 51-80.

Van Moffaert, K., Drugan, M. M. and Nowe, A. (2013) Scalarized multi-objective reinforcement learning: Novel design techniques. *2013 IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning , April 2013, Singapore*. pp. 191-199.

Veljanovska, K., Gacovski, Z. and Deskovski, S. (2012) Intelligent system for freeway ramp metering control. *6th IEEE International Conference on Intelligent Systems, September 2012, Sofia, Bulgaria.* pp. 279-282.

Veljanovska, K., M Bombol, K. and Maher, T. (2010) Reinforcement learning technique in multiple motorway access control strategy design. *PROMET-Traffic and Transportation, 22*(2), pp. 117-123.

Wang, J.J. and May, A. (1973) Computer model for optimal freeway on-ramp control. *Highway Research Record,* (469), pp. 16-25.

Wang, M.H. (1994) *Optimal ramp metering policies for nonrecurring congestion with uncertain incident duration*. Ph.D. thesis, Purdue University, USA.

Wang, X.J., Xi, X.M. and Gao, G.F. (2012) Reinforcement learning ramp metering without complete information. *Journal of Control Science and Engineering, 2012*, pp. 1-8.

Watkins, C. J. C. H. (1989) *Learning from delayed rewards*. Ph.D. thesis, University of Cambridge, UK.

Wattleworth, J. A. and Berry, D.S. (1965) Peak-period analysis and control of a freeway system-Some theoretical investigations. *Highway Research Board,* (89)*,* pp. 1-25.

Yafeng, Y., Hongchao, L. and Benouar, H. (2004) A note on equity of ramp metering. *7th International IEEE Conference on Intelligent Transportation Systems, October 2004, Washington, DC, USA*. pp. 497-502.

Yuan, L. S. and Kreer, J. B. (1971) Adjustment of freeway ramp metering rates to balance entrance ramp queues. *Transportation Research, 5*(2), pp. 127-133.

Yun, Z., Chen, Q. and Weili, H. (2010) Multi-objective reinforcement learning algorithm for MOSDMP in unknown environment. *8th World Congress on Intelligent Control and Automation, July 2010, Jinan, China*. pp. 3190-3194.

Zegeye, S. K., De Schutter, B., Hellendoorn, J., Breunesse, E. A. and Hegyi, A. (2012) A Predictive Traffic Controller for Sustainable Mobility Using Parameterized Control Policies. *IEEE Transactions on Intelligent Transportation Systems, 13*(3), pp. 1420-1429.

Zhang, G. and Wang, Y. (2013) Optimizing coordinated ramp metering: A preemptive hierarchical control approach. *Computer‐Aided Civil and Infrastructure Engineering, 28*(1), pp. 22-37.

Zhang, H., Ritchie, S. G. and Recker, W. W. (1996) Some general results on the optimal ramp control problem. *Transportation Research Part C: Emerging Technologies, 4*(2), pp. 51-69.

Zhang, H. M. and Recker, W. W. (1999) On optimal freeway ramp control policies for congested traffic corridors. *Transportation Research Part B: Methodological, 33*(6), pp. 417-436.

Zhang, H. M. and Ritchie, S. G. (1997) Freeway ramp metering using artificial neural networks. *Transportation Research Part C: Emerging Technologies, 5*(5), pp. 273-286.

Zhang, H. M. and Shen, W. (2010) Access control policies without inside queues: Their properties and public policy implications. *Transportation Research Part B: Methodological, 44*(8–9), pp. 1132-1147.

Zhang, L. and Levinson, D. (2005) Balancing efficiency and equity of ramp meters. *Journal of Transportation Engineering, 131*(6), pp. 477-481.

Zhao, D., Bai, X., Wang, F.-Y., Xu, J. and Yu, W. (2011) DHP method for ramp metering of freeway traffic. *IEEE Transactions on Intelligent Transportation Systems, 12*(4), pp. 990-999.

Zhaohui, Y. and Kaige, W. (2010) Multi-objective optimization of freeway traffic flow via a fuzzy reinforcement learning method. *3rd International Conference on Advanced Computer Theory and Engineering, August 2010, Chengdu, China*. pp. 530-534.

LIST OF ABBREVIATIONS

ACTM	Asymmetric cell transmission model
ADP	Adaptive dynamic programming
ALINEA	Asservissement Linéaire d'Entrée Autoroutière, i.e. Linear feedback control of a motorway on-ramp
ALINEA-C	ALINEA with continuous metering rates
ALINEA-D	ALINEA with discrete metering rates
ALINEA/Q	ALINEA combined with the queue management algorithm
AMOC	Advanced motorway optimal control
API	Application program interface
CTM	Cell transmission model
CFL	Courant–Friedrichs–Lewy condition
DLL	Dynamic link library
DP	Dynamic programming
DRL	Direct reinforcement learning
HATRIS	Highways agency traffic information system
IRL	Indirect reinforcement learning
JTDB	Journey time database
LWR	Lighthill-Whitham-Richards model
MDP	Markov decision process
MOO	Multi-objective optimisation problems
MORL	Multi-objective reinforcement learning
NC	Non-controlled situation
NE	Number of episodes
OAT	One-at-a-time sensitivity analysis
RAS	Ramp agent system

RAS-EQ	Ramp agent system with consideration of equity
RL	Reinforcement learning
SARSA	State-action-reward-state-action algorithm
SD	Standard deviation
TD	Temporal differential learning
TIMS	Traffic incident management system
TRADS	Traffic information database system
TTS	Total time spent
TTT	Total travel time
TWT	Total waiting time
TWTT	Total weighted travel time
VR	Variance of results
VSL	Variable speed limits
VMS	Variable message signs
WFTT	Weighted mainline travel time
WRD	Weighted on-ramp delay

APPENDIX A

SOURCE CODE LIST

A.1 Source Code of ACTM

```
//trafficflowmodel.h
//***************header file for ACTM*******************************

#ifndef TRAFFICFLOWMODEL_H_
#define TRAFFICFLOWMODEL_H_

#include <vector>
using namespace std;

class Cell
{
public:
    Cell(double li, double ve, double wa,
        double m_nmain, double q_c, double d_t);
    double q_in, q_out, delta_nmain, n_main;
    double max_qmain, max_dmain;
    double l, v, w;
    double t_simu;
    double lambda;

protected:
    double getMinThree (double a, double b, double c);
    double getMinTwo (double a, double b);
};

class CellNor:public Cell
{
public:
    CellNor(double li, double ve, double wa, double m_nmain, double
           q_c,double d_t);
    void traFlowNor(bool capa_drop, double w_next, double l_next,
                  double max_nmain_next, double n_main_next, double
                  theta_next, double m_r_next);
    void vehConsNor();
    void setInitialCell();
};

class CellOn: public Cell
{
public:
    CellOn(double li, double ve, double wa, double et, double th,
            double m_nmain, double q_c, double d_t);
    double d_on, m_r, delta_non, n_on;
    double c_action;
    double eta, theta;
```

```
    void traFlowOn(bool capa_drop, bool control, double w_next,
                 double l_next, double max_nmain_next, double
                 n_main_next, double theta_next, double m_r_next);
    void vehConsOn();
    void setInitialCell();
};

class CellOff: public Cell
{
public:
    CellOff(double li, double ve, double wa, double be,
        double m_nmain, double q_c, double d_t);
    double d_off;
    double beta;
    void traFlowOff(bool capa_drop, double w_next, double l_next,
                  double max_nmain_next, double n_main_next, double
                  theta_next, double m_r_next);
    void vehConsOff();
    void setInitialCell();
};

class CellOnf: public Cell
{
public:
    CellOnf(double li, double ve, double wa, double et, double th,
        double beta, double m_nmain, double q_c, double d_t);
    double d_on, m_r, delta_non, n_on;
    double c_action;
    double eta, theta, beta;
    void traFlowOnf(bool capa_drop, bool control, double w_next,
                   double l_next, double max_nmain_next,double
                   n_main_next, double theta_next, double m_r_next);
    void vehConsOnf();
    void setInitialCell();
};

#endif

//trafficflowmodel.cpp
//***************source file for ACTM*******************************

                //***************code 1**************

Cell::Cell(double li, double ve, double wa,
        double m_nmain, double q_c,double d_t)
{
    l = li;
    v = ve;
    w = wa;
    max_nmain = m_nmain;
    q_cap = q_c;
    t_simu = d_t;
    lambda = 0;
    q_in = q_out = delta_nmain =  0;
    n_main  = 0;
}

                //***************code 2**************
```

```
double Cell::getMinThree(double a, double b, double c)
{
    double d, min;
    d = (a<b?a:b);
    min = (d<c?d:c);
    return min;
}

                //**************code 3**************

double Cell::getMinTwo(double a, double b)
{
  if (a >= b) return b;
  else return a;
}

//functions for normal cell

                //**************code 4**************

CellNor::CellNor(double li, double ve, double wa,double m_nmain,
                 double q_c,double d_t)
           :Cell(li,ve,wa,m_nmain,q_c,d_t)
{
}
                //**************code 5**************

void CellNor::traFlowNor(bool capa_drop, double w_next, double
                         l_next, double max_nmain_next,double
                         n_main_next, double theta_next, double
                         m_r_next)
{
    //determine capacity drop
    if (capa_drop == true) lambda = 0.9;
    else lambda = 1;

    //calculate mainline traffic flow
    if (n_main/l <= den_crit)
    {
       if (n_main_next/l_next <= 60)
       q_out = v/l*n_main;
       if (n_main_next/l_next > 60)
       q_out = getMinTwo(v/l*n_main, w_next/l_next
               *(max_nmain_next*l_next-n_main_next-
               theta_next*m_r_next*t_simu));
    }
    if (n_main/l > den_crit)
    {
       if (n_main_next/l_next <= den_crit)
       q_out = lambda*q_cap;
       if (n_main_next/l_next > den_crit)
       q_out = w_next/l_next*(max_nmain_next*l_next-n_main_next-
               theta_next*m_r_next*t_simu);
    }
}
                //**************code 6**************

void CellNor::vehConsNor()
{
```

```
    delta_nmain = (q_in - q_out) * t_simu;
    n_main += delta_nmain;
    if (n_main < 0) n_main = 0;
}
                //**************code 7**************

void CellNor::setInitialCell()
{
    q_in = q_out = delta_nmain = 0;
    n_main = 0;
}


//functions for on-ramp cell

                //**************code 8**************

CellOn::CellOn(double li, double ve, double wa, double et, double
             th, double m_nmain, double q_c,double d_t)
        :Cell(li, ve, wa, m_nmain, q_c, d_t)
{
    d_on = m_r = delta_non = n_on = c_action = 0;
    eta = et;
    theta = th;
}
                //**************code 9**************

void CellOn::traFlowOn(bool capa_drop,bool control,double w_next,
                       double l_next, double max_nmain_next, double
                       n_main_next, double theta_next, double
                       m_r_next)
{
    //determine capacity drop
    if (capa_drop == true) lambda = 0.9;
    else lambda = 1;

    //calculate mainline flow
    if (n_main/l <= den_crit)
    {
       if (n_main_next/l_next <= den_crit)
       q_out = v/l*(n_main + theta*m_r*t_simu);
       if (n_main_next/l_next > den_crit)
       q_out = getMinTwo(v/l*(n_main + theta*m_r*t_simu),
               w_next/l_next*(max_nmain_next*l_next-n_main_next-
               theta_next*m_r_next*t_simu));
    }
    if (n_main/l > den_crit)
    {
       if (n_main_next/l_next <= den_crit)
       q_out = lambda*q_cap;
       if (n_main_next/l_next > den_crit)
       q_out = w_next/l_next*(max_nmain_next*l_next-n_main_next-
              theta_next*m_r_next*t_simu);
    }

    //calculate on-ramp flow
    if (control==true)
    {
```

```
        m_r = getMinThree((n_on+d_on*t_simu)/t_simu,
              eta*(max_nmain*l-n_main)/t_simu, c_action/t_simu);
    }
    if (control==false)
    {
        m_r = getMinTwo((n_on+d_on*t_simu)/t_simu,
               eta*(max_nmain*l-n_main)/t_simu);

    }
}
                //**************code 10**************

void CellOn::vehConsOn()
{
    //mainline conservation
    delta_nmain = (q_in + m_r - q_out) * t_simu;
    n_main += delta_nmain;

    //on-ramp conservation
    delta_non = (d_on - m_r) * t_simu;
    n_on += delta_non;

    if (n_main < 0) n_main = 0;
    if (n_on < 0) n_on = 0;
}
                //**************code 11**************

void CellOn::setInitialCell()
{
    q_in = q_out = delta_nmain = 0;
    d_onm = m_r = delta_non = n_on = c_action = 0;
    n_main = 0;
}

//functions for celloff

                //**************code 12**************

CellOff::CellOff(double li, double ve, double wa, double be,
                 double m_nmain, double q_c,double d_t)
        :Cell(li, ve, wa, m_nmain, q_c, d_t)
{
    d_off = 0;
    beta = be;
}
                //**************code 13**************

void CellOff::traFlowOff(bool capa_drop, double w_next, double
                         l_next, double max_nmain_next, double
                         n_main_next, double theta_next, double
                         m_r_next)
{
    //determine capacity drop
    if (capa_drop == true) lambda = 0.9;
    else lambda = 1;

    //calculate mainline flow
    if (n_main/l <=  den_crit)
    {
```

```
        if (n_main_next/l_next <=  den_crit)
        q_out = (1-beta)*v/l*n_main;
        if (n_main_next/l_next >  den_crit)
        q_out = getMinTwo((1-beta)*v/l*n_main,
                w_next/l_next*(max_nmain_next*l_next-n_main_next-
                theta_next*m_r_next*t_simu));
    }
    if (n_main/l >  den_crit)
    {
        if (n_main_next/l_next <=  den_crit)
        q_out = lambda*q_cap;
        if (n_main_next/l_next >  den_crit)
        q_out = w_next/l_next*(max_nmain_next*l_next-n_main_next-
        theta_next*m_r_next*t_simu);
    }
}
                //**************code 14**************

void CellOff::vehConsOff()
{
    delta_nmain = (q_in - q_out/(1-beta)) * t_simu;
    n_main += delta_nmain;
    if (n_main < 0) n_main = 0;
}
                //**************code 15**************

void CellOff::setInitialCell()
{
    q_in = q_out = delta_nmain =  0;
    n_main = 0;
    d_off = 0;
}


//functions for cellonf

                //**************code 16**************

CellOnf::CellOnf(double li, double ve, double wa, double et, double
                 th, double be, double m_nmain, double q_c, double
                 d_t)
      :Cell(li, ve, wa, m_nmain, q_c, d_t)
{
    d_onm = m_r = delta_non = n_on = c_action = 0;
    eta = et;
    theta = th;
    beta = be;
}
                //**************code 17**************

void CellOnf::traFlowOnf(bool capa_drop, bool control, double
                         w_next, double l_next, double
                         max_nmain_next, double n_main_next, double
                         theta_next, double m_r_next)
{
    //determine capacity drop
    if (capa_drop == true) lambda = 0.9;
    else lambda = 1;
```

```
    //calculate mainline flow
     if (n_main/l <= den_crit)
     {
        if (n_main_next/l_next <= den_crit)
        q_out = (1-beta)*v/l*(n_main + theta*m_r*t_simu);
        if (n_main_next/l_next > den_crit)
        q_out = getMinTwo((1-beta)*v/l*(n_main + theta*m_r*t_simu),
                w_next/l_next*(max_nmain_next*l_next-n_main_next-
               theta_next*m_r_next*t_simu));
     }
     if (n_main/l > den_crit)
     {
        if (n_main_next/l_next <= den_crit)
        q_out = lambda*q_cap;
        if (n_main_next/l_next > den_crit)
        q_out = w_next/l_next*(max_nmain_next*l_next-n_main_next-
                theta_next*m_r_next*t_simu);
     }

    //calculate on-ramp flow
     if (control==true)
     {
         m_r = getMinThree((n_on+d_on*t_simu)/t_simu,
          eta*(max_nmain*l-n_main)/t_simu, c_action/t_simu);
     }
     if (control==false)
     {
         m_r = getMinTwo((n_on+d_on*t_simu)/t_simu,
          eta*(max_nmain*l-n_main)/t_simu);
     }
}
                //**************code 18**************

void CellOnf::vehConsOnf()
{
     //mainline conservation
     delta_nmain = (q_in + m_r - q_out/(1-beta)) * t_simu;
     n_main += delta_nmain;

    //on-ramp conservation
     delta_non = (d_on - m_r) * t_simu;
     n_on += delta_non;

     if (n_main < 0) n_main = 0;
     if (n_on < 0) n_on = 0;
}
                //**************code 19**************

void CellOnf::setInitialCell()
{
     q_in = q_out = delta_nmain =  0;
     d_onm = m_r = delta_non = n_on = c_action = 0;
     n_main = 0;
}
```

A.2 Source Code of RampAgent

```
//rampagent.h
//**************header file for RampAgent************************

#ifndef RAMPAGENT_H_
#define RAMPAGENT_H_

#include "objective.h"
#include <iostream>
#include <fstream>
#include <vector>
#include <string.h>
#include <stdio.h>
using namespace std;

class RampAgent
{
public:
    RampAgent(double h_nmain,double l_nmain,double d_nmain,double
              h_non,double l_non,double d_non,double h_qin,double
              l_qin,double d_qin,double h_don,double l_don,double
              d_don);
    RampAgent();

    double n_main, n_on, q_in, d_on;
    double delta_nmain, delta_non, delta_qin, delta_don,
           higher_nmain, higher_non, higher_qin, higher_don,
           lower_nmain, lower_non, lower_qin, lower_don;
    double epsilon;
    int action[9];
    int pr_state;
    int cu_state;
    int pr_action;
    int cu_action;
    void setObjective(int o_number);
    void deleteObjective(int o_number);
    void setInitialSA();
    vector<Objective*> obj;
    int state_number, action_number;
    int state_main, state_ramp;
    int greedy_action;

protected:
    void getState();
    void getActionEG(int upperBound, int lowerBound);
    long myrandom(long n);
};

class SORampAgent: public RampAgent
{
public:
    SORampAgent(double h_nmain,double l_nmain,double d_nmain,double
                h_non,double l_non,double d_non,double h_qin,double
                l_qin,double d_qin,double h_don,double l_don,double
                d_don);
    SORampAgent();
    void startStateTransition();
    void startStateTransitionFA();
```

```
    void inputRQ();
    void outputRQ(ofstream& out);

private:
    void getGreedyAction();
};

class MORampAgent: public RampAgent
{
public:
    MORampAgent(double h_nmain,double l_nmain,double d_nmain,double
                h_non,double l_non,double d_non,double h_qin,double
                l_qin,double d_qin,double h_don,double l_don,double
                d_don, double o_number);
    MORampAgent();
    double obj_number;
    void startStateTransition(double w_1);
    void setSQSize();
    void inputRQ();
    void outputRQ();

private:
    double max_SQ;
    vector<vector <double> > S_Q;
    void getSQValue(double w_1);
    void getGreedyActionSQ();
};

#endif

//rampagent.cpp
//***************source file for RampAgent**************************

                //***************code 1**************

RampAgent::RampAgent(double h_nmain,double l_nmain,double
                     d_nmain,double h_non,double l_non,double
                     d_non,double h_qin,double l_qin,double
                     d_qin,double h_don,double l_don,double d_don)
{
    int temp_arr[9] = {2,3,4,5,6,7,8,9,10};
    memcpy(action,temp_arr,sizeof(temp_arr));
    higher_nmain = h_nmain;
    lower_nmain = l_nmain;
    delta_nmain = d_nmain;
    higher_non = h_non;
    lower_non = l_non;
    delta_non = d_non;
    higher_qin = h_qin;
    lower_qin = l_qin;
    delta_qin = d_qin;
    higher_don = h_don;
    lower_don = l_don;
    delta_don = d_don;
    action_number = 9;
    state_number = (ceil((h_nmain-l_nmain)/d_nmain+2))
                  *(ceil((h_non-l_non)/d_non+2))
                  *(ceil((h_qin-l_qin)/d_qin+2))
                  *(ceil((h_don-l_don)/d_don+2));
```

```
}

                //**************code 2**************

void RampAgent::setObjective(int o_number)
{
    for (int i=0; i< o_number; i++)
    {
        Objective *ob;
        ob = new Objective;
        obj.push_back(ob);
        obj[i]->setSize(state_number, action_number);
    }
}

                //**************code 3**************

void RampAgent::deleteObjective(int o_number)
{
    for (int i=0; i< o_number; i++)
    {
        delete obj[i];
    }
}

                //**************code 4**************

void RampAgent::setInitialSA()
{
    pr_state = 0;
    pr_action = 8;
    cu_state = 0;
    cu_action = 8;
    n_main = 0;
    n_on = 0;
    greedy_action = 0;
    q_in = d_on = 0;
}

                //**************code 5**************

void RampAgent::getState()
{
    //state for mainline vehicles
    if (n_main <= lower_nmain) state_nmain = 0;
    if (n_main > lower_nmain && n_main <= higher_nmain )
        state_nmain = ceil((n_main-lower_nmain)/delta_nmain);
    if (n_main > higher_nmain )
        state_nmain = ceil((higher_nmain-
         lower_nmain)/delta_nmain+1);

    //state for mainline inflow
    if (q_in <= lower_qin) state_qin = 0;
    if (q_in > lower_qin && q_in <= higher_qin)
        state_qin = ceil((q_in-lower_qin)/delta_qin);
    if (q_in > higher_qin)
        state_qin = ceil((higher_qin-lower_qin)/delta_qin+1);
```

```
    //state for on-ramp queue
    if (n_on <= lower_non) state_non = 0;
    if (n_on > lower_non && n_on <= higher_non)
        state_non = ceil((n_on-lower_non)/delta_non);
    if (n_on > higher_non)
        state_non = ceil((higher_non-lower_non)/delta_non+1);

    //state for on-ramp demand
    if (d_on <= lower_don) state_don = 0;
    if (d_on > lower_don && d_on <= higher_don)
        state_don = ceil((d_on-lower_don)/delta_don);
    if (d_on > higher_don)
        state_don = ceil((higher_don-lower_don)/delta_don+1);

    //integrated state
    cu_state =
             state_qin_number*state_non_number*state_don_number*sta
             te_nmain+state_non_number*state_don_number*state_qin
            + state_don_number*state_non + state_don;
}

                //**************code 6**************

void RampAgent::getActionEG(int upperBound, int lowerBound)
{
    int range = (upperBound - lowerBound)+1;
    int a;
    //generate a random number between 0 and 99
    a = rand()%100;

    //find the non-greedy action with the probability epsilon/100
    if(a<epsilon)
    {
        do {cu_action = lowerBound + int(range * rand() / (RAND_MAX
            + 1.0));
            }
        while(cu_action == greedy_action);
    }
    else
    {
        cu_action = greedy_action;
    }
}


//source code for SORampAgent

                //**************code 7**************

SORampAgent::SORampAgent(double h_nmain,double l_nmain,double
                    d_nmain,double h_non,double l_non,double d_non,
                    double h_qin,double l_qin,double d_qin,double
                    h_don,double l_don,double d_don)
:RampAgent(h_nmain,l_nmain,d_nmain,h_non,l_non,d_non,h_qin,l_qin,d_
qin,h_don,l_don,d_don)
{
    setObjective(1);
}
```

```
                //**************code 8**************

void SORampAgent::startStateTransition()
{
    getState();
    obj[0]->getReward(pr_state, pr_action);

    getGreedyAction();
    getActionEG(8,0);
    obj[0]->getQValue(pr_state, pr_action, cu_state, greedy_action);
    pr_state = cu_state;
    pr_action = cu_action;
}

                //**************code 9**************

void SORampAgent::getGreedyAction()
{
    int greedy = 0;
    //find the greedy action with the maximum Q
    for (int i=0; i < action_number; i++)
    {
        if((obj[0]->Q[cu_state][i]) > (obj[0]->Q[cu_state][greedy]))
        greedy = i;
    }
    greedy_action = greedy;
}

                //**************code 10**************

void SORampAgent::inputRQ()
{
    ifstream in("M:\\SORQ.txt");//open the file
    //read data from the file
    for(int i1 = 0; i1 < state_number; i1++){
        for(int j1 = 0; j1 < action_number; j1++){
            in >> obj[0]->Q[i1][j1];
        }
    }
    in.close();
}

                //**************code 11**************

void SORampAgent::outputRQ(ofstream& out)
{
   for(int i1 = 0; i1 < state_number; i1++){
       for(int j1 = 0; j1 < action_number; j1++){
           out<<obj[0]->Q[i1][j1]<<" ";
        }
        out<<endl;
        if (i1 == state_number-1) {
        out<<endl;
        }
     }
}

//source code for MORampAgent
```

```
                //**************code 12**************

MORampAgent::MORampAgent(double h_nmain,double l_nmain,double
                         d_nmain,double h_non,double l_non,double
                         d_non,double h_qin,double l_qin,double
                         d_qin,double h_don,double l_don,double
                         d_don,double o_number)
:RampAgent(h_nmain,l_nmain,d_nmain,h_non,l_non,d_non,h_qin,l_qin,d_
qin,h_don,l_don,d_don)
{
    setSQSize();
    obj_number = o_number;
    setObjective(obj_number);
}

                //**************code 13**************

void MORampAgent::setSQSize()
{
    S_Q.resize(state_number,vector<double>(action_number,0));
}

                //**************code 14**************

void MORampAgent::startStateTransition(double w_1)
{
    getState();
    for (int i=0; i< obj_number; i++)
    {
        obj[i]->getReward(pr_state, pr_action);
    }
    getGreedyActionSQ();
    getActionEG(8,0);

    for (int i1=0; i1< obj_number; i1++)
    {
        obj[i1]->getQValue(pr_state, pr_action, cu_state,
cu_action);
    }
    getSQValue(w_1);
    pr_state = cu_state;
    pr_action = cu_action;
}

                //**************code 15**************

void MORampAgent::inputRQ()
{
    ifstream in("M:\\MORQ.txt");//open the file
    //read data from the file
    for(int i1 = 0; i1 < state_number; i1++){
        for(int j1 = 0; j1 < action_number; j1++){
            in >> obj[0]->Q[i1][j1];
        }
    }
    for(int i2 = 0; i2 < state_number; i2++){
        for(int j2 = 0; j2 < action_number; j2++){
            in >> obj[1]->Q[i2][j2];
        }
```

```
    }
    for(int i3 = 0; i3 < state_number; i3++){
        for(int j3 = 0; j3 < action_number; j3++){
            in >> S_Q[i3][j3];
        }
    }
    in.close();
}
                //**************code 16**************

void MORampAgent::outputRQ()
{
    ofstream out("M:\\MORQ.txt");
    for(int i2 = 0; i2 < state_number; i2++){
        for(int j2 = 0; j2 < action_number; j2++){
            out<<obj[0]->Q[j2][j2]<<" ";
        }
        out<<endl;
        if (i2 == state_number-1) {
        out<<endl;
        }
     }
     for(int i3 = 0; i3 < state_number; i3++){
        for(int j3 = 0; j3 < action_number; j3++){
            out<<obj[1]->Q[i3][j3]<<" ";
        }
        out<<endl;
        if (i3 == state_number-1) {
        out<<endl;
        }
     }
     for(int i4 = 0; i4 < state_number; i4++){
        for(int j4 = 0; j4 < action_number; j4++){
            out<<S_Q[i4][j4]<<" ";
        }
        out<<endl;
        if (i4 == state_number-1) {
        out<<endl;
        }
     }
     out.close();
}
                //**************code 17**************

void MORampAgent::getGreedyActionSQ()
{
    int greedy = 0;
    for (int i=0; i < action_number; i++)
    {
        if(S_Q[cu_state][i] > S_Q[cu_state][greedy])
        greedy = i;
    }
    greedy_action = greedy;
}
                //**************code 18**************

void MORampAgent::getSQValue(double w_1)
{
    S_Q[pr_state][pr_action] =
```

```
                w_1 * obj[0]->Q[pr_state][pr_action] + (1-w_1) *
                obj[1]->Q[pr_state][pr_action];
}

                //***************code 19**************

void SORampAgent::startStateTransition()
{          //basic Q-learning
    getState();
    obj[0]->getReward(pr_state, pr_action);
    getGreedyAction();
    getActionEG(8,0);
    obj[0]->getQValue(pr_state, pr_action, cu_state,
greedy_action);
    pr_state = cu_state;
    pr_action = cu_action;
         //model learning
    pr_state_plan = pr_state;
    pr_action_plan = pr_action;
    cu_state_plan = cu_state;
    plan_step = control_step;
    n_main_plan = n_main;
    n_on_plan = n_on;
    capaDistribution(q_out);
    q_in_plan = (qin[0]+qin[1]+qin[2]+qin[3]+qin[4])/5;
    d_on_plan = (don[0]+don[1]+don[2]+don[3]+don[4])/5;
          //planning
    for (plan_step = 0; plan_step <= plan_max; plan_step++)
    {
        q_out_plan = q_cap*capa_redu;
        getState();
        obj[0]->getReward(pr_state_plan, pr_action_plan);
        getGreedyAction();
        getActionEG(8,0);
        obj[0]->getQValue(pr_state_plan, pr_action_plan,
cu_state_plan, greedy_action);
        pr_state_plan = cu_state_plan;
        pr_action_plan = cu_action_plan;
        if ((plan_step == control_step+9)||((q_out <= q_bound) &&
(plan_step+1 >= incident_step))
        plan_max = plan_step+1;
    }
}
```

A.3 Source Code of Objective

```
//objective.h
//**************header file for Objective*************************

#ifndef OBJECTIVEFUNCTION_H_
#define OBJECTIVEFUNCTION_H_

#include <iostream>
#include <fstream>
#include <vector>
#include <string.h>
#include <stdio.h>
using namespace std;

class Objective
{
public:
    void setSize(int state_number, int action_number);
    double reward_value, reward, min_reward, max_reward;
    double max_reward;
    double min_reward;
    double gamma;
    double alpha;

    vector<vector <double> > ad_alpha;
    vector<vector <double> > Q;
    void getReward(int pr_state, int pr_action);
    void getQValue(int pr_state, int pr_action, int cu_state, int
                   cu_action);
    void getAdQValue(int pr_state, int pr_action, int cu_state, int
                     cu_action);
    void countAlpha(int pr_state, int pr_action);
    void resizeAlpha(int state_number, int action_number);
};

#endif

//objective.cpp
//**************source file for Objective*************************

                //**************code 1**************

void Objective::setSize(int state_number, int action_number)
{
    ad_alpha.resize(state_number,vector<double>(action_number,0));
    Q.resize(state_number,vector<double>(action_number,0));
}

                //**************code 2**************

void Objective::resizeAlpha(int state_number, int action_number)
{
    ad_alpha.resize(state_number,vector<double>(action_number,0));
}

                //**************code 3**************
```

```
void Objective::getReward(int pr_state, int pr_action)
{
    reward = (reward_value-min_reward)/(max_reward-min_reward);
}

                //**************code 4**************

void Objective::countAlpha(int pr_state, int pr_action)
{
    ad_alpha[pr_state][pr_action]  =  ad_alpha[pr_state][pr_action]+1;
}

                //**************code 5**************

void Objective::getQValue(int pr_state, int pr_action, int cu_state,
int cu_action)
{
    Q[pr_state][pr_action] = Q[pr_state][pr_action]+
        alpha * (reward + gamma * Q[cu_state][cu_action]-
Q[pr_state][pr_action]);
}

                //**************code 6**************

void Objective::getAdQValue(int pr_state, int pr_action, int
cu_state, int cu_action)
{
    countAlpha(pr_state, pr_action);
    Q[pr_state][pr_action] =
            Q[pr_state][pr_action]+(1/ad_alpha[pr_state][pr_action])
            *(reward + gamma * Q[cu_state][cu_action]-
            Q[pr_state][pr_action]);
}
```

APPENDIX B
CALIBRATION OF ALINEA

B.1 Calibration in Single-ramp Case

It has been mentioned in (Gomes and Horowitz 2003) that, to make ALINEA work under ACTM, two parameters should satisfy: $K_R \in (0, 2(2-v_{nor})]$ and $\hat{\rho} \in [q_{dis}/v, \rho_{crit}]$. In this study, the normalised free-flow speed v_{nor} can be calculated by: $v_{nor} = (v \cdot T_s)/l \approx 0.833$, and the discharge rate is $q_{dis} = 0.9 \cdot \rho_{crit}$. Thus, two parameters should be in the range: $K_R \in (0, 2.3]$ and $\hat{\rho} \in [0.9 \cdot \rho_{crit}, \rho_{crit}]$. Within these two ranges, two parameters of ALINEA can be calibrated to obtain the minimum TTS of the controlled cell 2.

ALINEA-C

Tables B.1 and B.2 show the calibration results for ALINEA-C. To calibrate K_R, the target density $\hat{\rho}$ is set as the critical value ρ_{crit} = 20 veh/lane/km, which is not changed during the test. The optimal K_R found in Table B.1 will be used in the calibration of $\hat{\rho}$.

Table B.1: Calibration of K_R for ALINEA-C

K_R	0.10	0.20	0.25	0.30	0.35	0.40	0.50	1.00	1.50	2.00	2.30
TTS of cell 2	3828	3689	3775	3664	3760	7218	8155	8732	9360	9232	9121

Table B.2: Calibration of $\hat{\rho}$ for ALINEA-C

$\hat{\rho}$	ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.90 ρ_{crit}
TTS of cell 2	3664	4063	4572	5202	5968	6882	7958	9164	10494	11923	13362

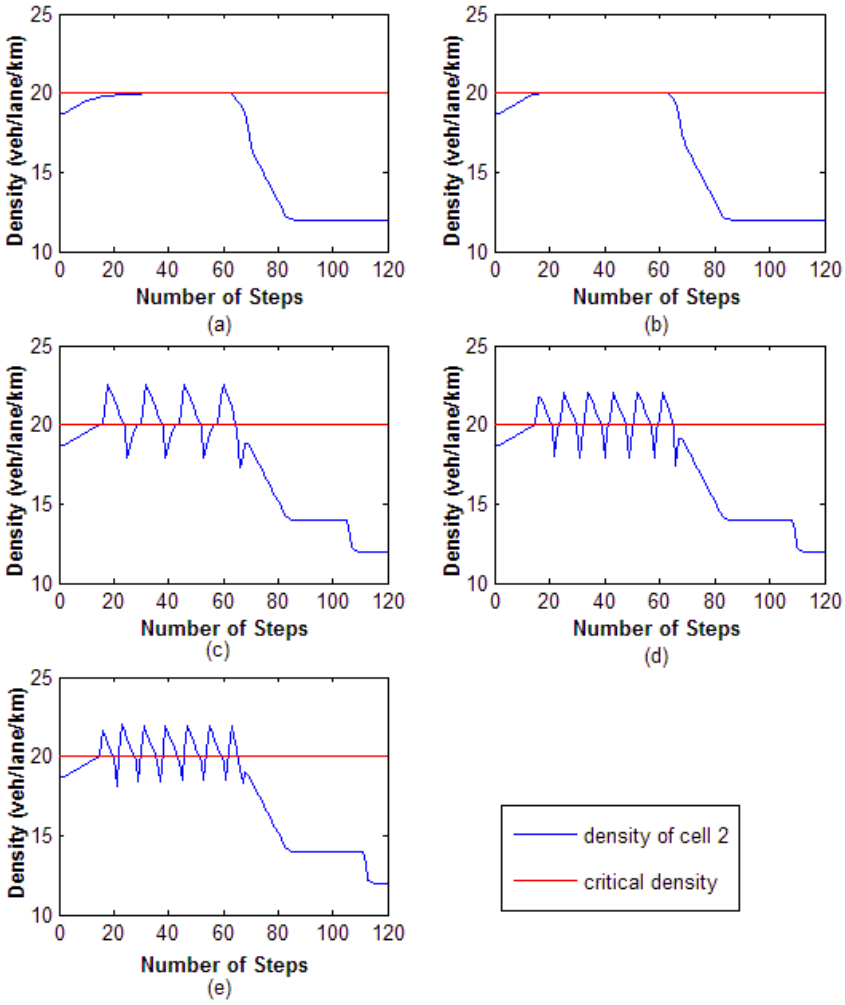


Figure B.1: density of cell 2 under different K_R: (a) 0.1, (b) 0.3, (c) 0.5, (d) 1, (e) 2

As shown in Figure B.1, with smaller K_R (below 0.3), cell density can be steadily maintained around the critical density, while it takes more time

(around 40 steps) than larger K_R to reach this value. When K_R is above 0.3, the critical density can be reached faster (20 steps), but the algorithm is getting unstable. An optimal K_R which is fast and stable can be found at K_R = 0.3. With this parameter value, ALINEA-C can obtain the minimum TTS of 3664 veh.min. $\hat{\rho}$ is better set as the critical value (20 veh/lane/km), smaller values lead to lower outflow and thus have longer TTS.

ALINEA-D

ALINEA-D can be calibrated by the same method and the calibration results can be found in Tables B.3 and B.4. The same as ALINEA-C, K_R = 0.3 and $\hat{\rho} = \rho_{crit}$ are also optimal for ALINEA-D.

Table B.3: Calibration of $\hat{\rho}$ for ALINEA-D

$\hat{\rho}$	ρ_{crit}	**0.99** ρ_{crit}	**0.98** ρ_{crit}	**0.97** ρ_{crit}	**0.96** ρ_{crit}	**0.95** ρ_{crit}	**0.94** ρ_{crit}	**0.93** ρ_{crit}	**0.92** ρ_{crit}	**0.91** ρ_{crit}	**0.90** ρ_{crit}
TTS of cell 2	3917	4963	4969	6509	8596	8657	8716	8776	8857	8896	11664

Table B.4: Calibration of K_R for ALINEA-D

K_R	**0.10**	**0.20**	**0.25**	**0.30**	**0.35**	**0.40**	**0.50**	**1.00**	**1.50**	**2.00**	**2.30**
TTS of cell 2	8676	4962	4946	3917	3917	3917	3917	3917	9283	9261	9166

B.2 Calibration in Multi-ramp Case

In the first demand scenario, only one on-ramp may cause congestion which is similar to the single-agent case. Calibrated parameters from B.1 also work well in this test. However, under demand profile 2 and 3, these parameter values are no longer effective and should be recalibrated.

B.2.1 Demand profile 2

Figure B.2 shows mainline densities in three controlled cells (cell 6, 9 and 12) under demand profile 2 with parameters calibrated from B.1. For cell 6 and 9, mainline density can be successfully maintained around the critical value (20 veh/lane/km), while the density of cell 12 is not effectively controlled which even reaches 45 veh/lane/km in some time steps. Thus, the calibration under demand profile 2 will be mainly focused on the controller (controller 3) that controls cell 12 (corresponding to on-ramp 3).

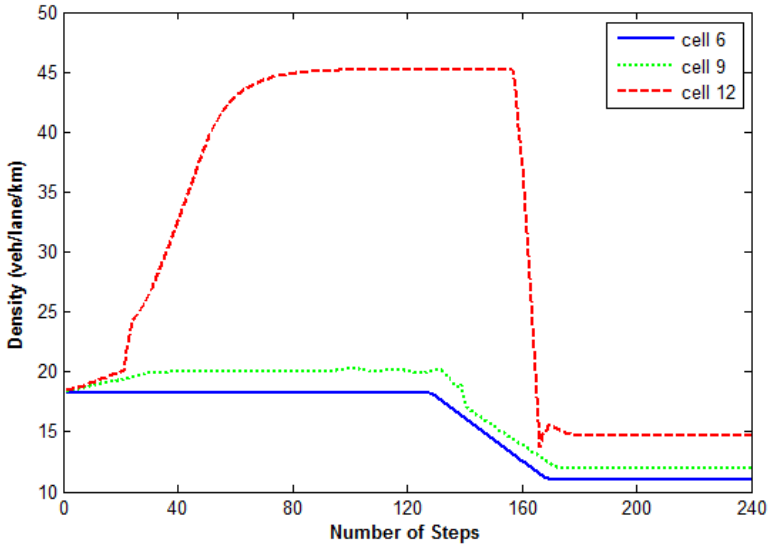


Figure B.2: Cell density under demand 2

ALINEA-C

Firstly, different values of $\hat{\rho}_3$ are tested with fixed $K_{R,3}$ = 0.3 (optimal value in B.1). The calibration result can be seen from Table B.5, where $\hat{\rho}$ = 0.98 ρ_{crit} is found to be better than other values.

Table B.5: Calibration of $\hat{\rho}_3$ for ALINEA-C (demand 2)

$\hat{\rho}_3$		ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.9 ρ_{crit}
TTS	**Sect 3**	19147	18801	9733	10888	12186	13551	14966	15580	16090	16551	16965
	Network	37769	37416	28428	29583	30868	32202	33585	34186	34685	35135	35540

When $K_{R,3}$ is less than 0.5, the algorithm performance is very stable with almost the same network TTS. The same as B.1, $K_{R,3}$ = 0.3 will be chosen as the optimal parameter value.

Table B.6: Calibration of $K_{R,3}$ for ALINEA-C (demand 2)

$K_{R,3}$	0.10	0.20	0.25	0.30	0.35	0.40	0.50	1.00	1.50	2.00	2.30
Network TTS	28428	28428	28428	28428	28428	28429	28429	32494	33668	33636	33846

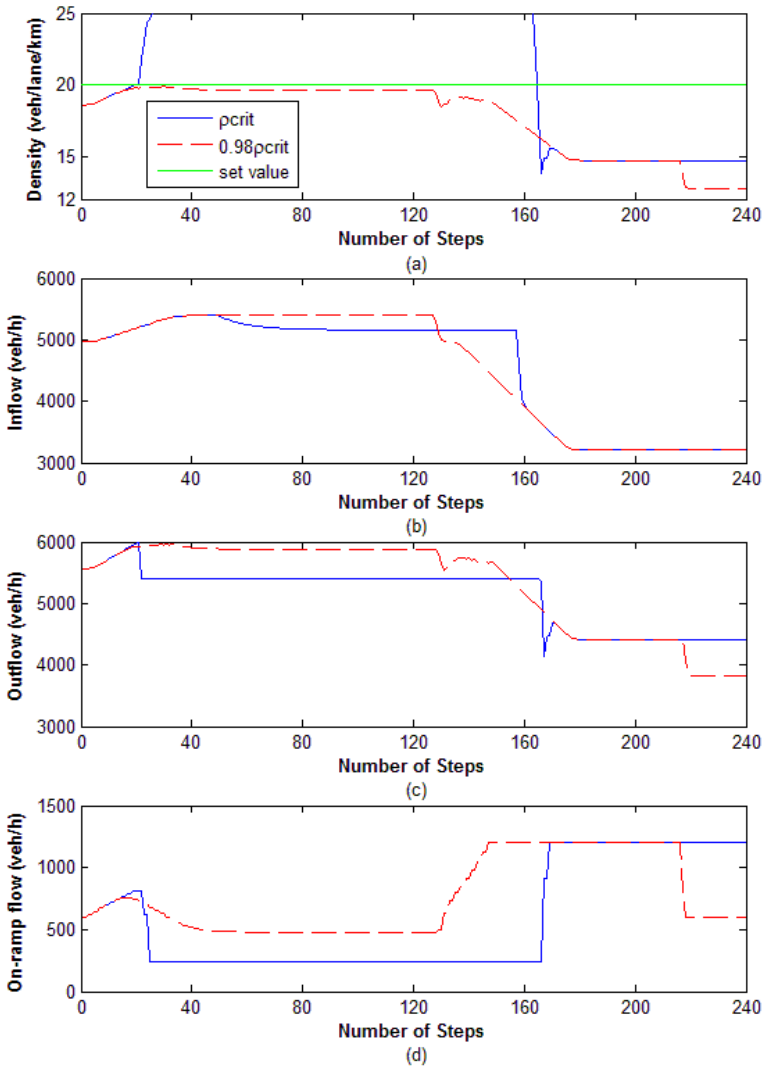
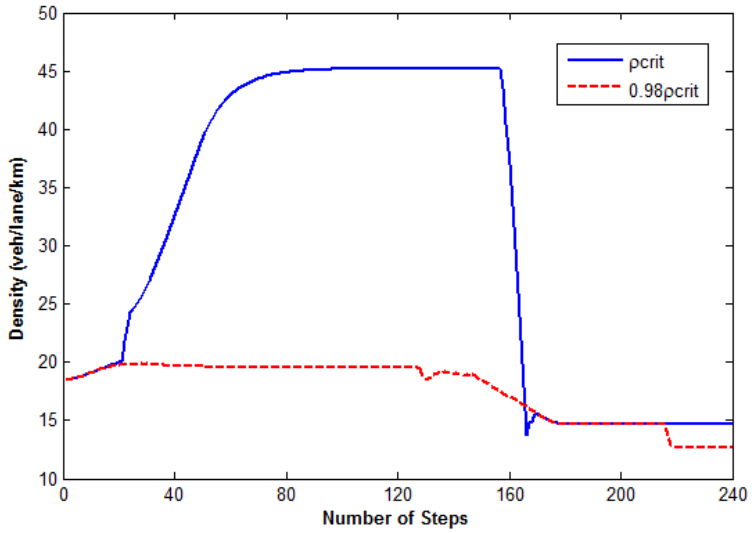


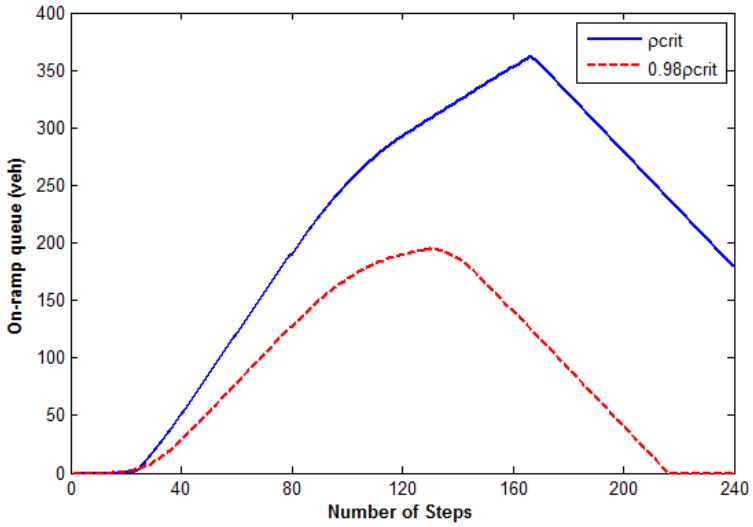
Figure B.3: Comparison of: (a) cell densities, (b) cell inflows, (c) cell outflows, (d) on-ramp flows

A comparison of $\hat{\rho}_3 = \rho_{crit}$ and $\hat{\rho}_3 = 0.98\,\rho_{crit}$ can be seen from Figures B.3 and B.4, which can be used to explain why slight reduction of $\hat{\rho}_3$ sometimes leads to a drastic change of TTS. As shown in Figure B.3 (a), when $\hat{\rho}_3$ is set as the critical density (20 veh/lane/km), the controller will try to reach this value and sometimes may make the mainline density a little bit higher than 20 veh/lane/km (around the 20th step). Once mainline density exceeds the

critical value, congestion will happen and the cell outflow will drop to the 5400 veh/h because of capacity drop (see Figure B.3 (c)). Although the cell inflow will reduce because of congestion, the sum of the minimum controlled on-ramp flow (240 veh/h) and cell inflow (5160 veh/h) is still not smaller than 5400 veh/h. Thus, the mainline density cannot be eliminated as shown in Figure B.4 (a). In the meanwhile, this minimum on-ramp flow causes a long vehicle queue on the on-ramp (Figure B.4 (b)). Both of the mainline congestion and long on-ramp queue lead to a high TTS. When $\hat{\rho}$ is set as a lower value such as 0.98 ρ_{crit}, the mainline density will never exceed the critical value, and accordingly, the outflow of cell 12 can keep at a higher level around 5900 veh/h. Therefore, as shown in Figure B.4 (a) and (b), the mainline congestion can be completely eliminated, and a much shorter queue can be found on the on-ramp. Thus, when $\hat{\rho}_3$ = 0.98 ρ_{crit}, the TTS can be greatly reduced.



(a)



(b)

Figure B.4: Comparison of: (a) cell densities, (b) on-ramp queues

ALINEA-D

Following the same process introduced above, the calibration results of ALINEA-D can be obtained and illustrated in Tables B.7 and B.8. $\hat{\rho}_3$=0.97 ρ_{crit} and $K_{R,3}$= 0.3 are optimal parameter values.

Table B.7: Calibration of $\hat{\rho}_3$ for ALINEA-D (demand 2)

$\hat{\rho}_3$		ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.9 ρ_{crit}
TTS	Sect 3	18500	17932	17932	8870	9009	11599	11771	14226	14530	17099	17539
	Network	37818	37257	37257	28355	28494	31074	31242	33639	33936	36445	36876

Table B.8 Calibration of $K_{R,3}$ for ALINEA-D (demand 2)

$K_{R,3}$	0.10	0.20	0.25	0.30	0.35	0.40	0.50	1.00	1.50	2.00	2.30
Network TTS	29104	37291	37274	28355	28378	28385	28461	29416	29407	36853	37859

B.2.2 Demand profile 3

Under demand profile 3, when parameters calibrated from B.1 are used, both of cell 6 and 9 (corresponding to controller 2 and 3) are not well controlled (see Figure B.5). Therefore, parameters of two controllers ($\hat{\rho}_2$, $K_{R,2}$, $\hat{\rho}_3$, $K_{R,3}$) controlling these two cells should be recalibrated.

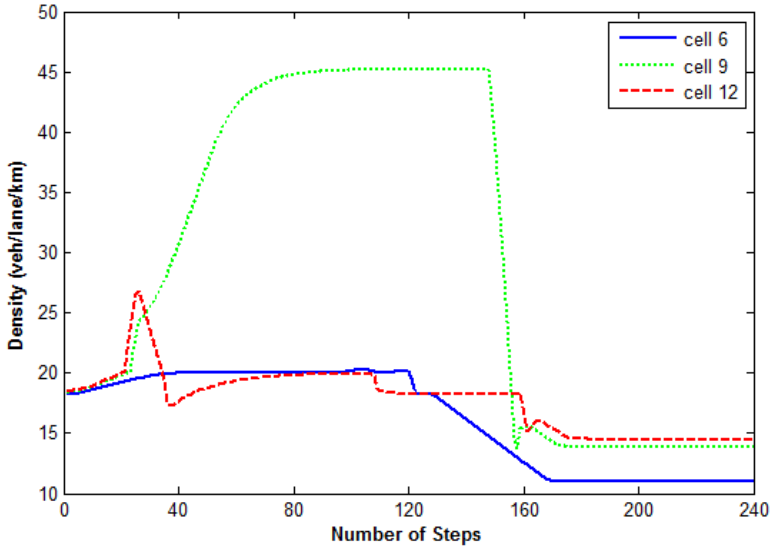


Figure B.5: Cell density under demand 3

ALINEA-C

Firstly, $\hat{\rho}_2$ is calibrated with all other parameters fixed, i.e. $K_{R,2}$ = 0.3, $\hat{\rho}_3 = \rho_{crit}$ and $K_{R,3}$ = 0.3. When the optimal $\hat{\rho}_2$ is found, $\hat{\rho}_2$ will be set as this value in the calibration of $\hat{\rho}_3$. Tables B.9 and B.10 present the results for $\hat{\rho}_2$ and $\hat{\rho}_3$, from which we can see that both $\hat{\rho}_2$ and $\hat{\rho}_3$ should be 0.98 ρ_{crit}.

Table B.9: Calibration of $\hat{\rho}_2$ for ALINEA-C (demand 3)

$\hat{\rho}_2$	ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.9 ρ_{crit}
TTS of Sect 2	19147	18801	9733	10888	12186	13551	14966	15580	16090	16551	16965

Table B.10: Calibration of $\hat{\rho}_3$ for ALINEA-C (demand 3)

$\hat{\rho}_3$		ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.9 ρ_{crit}
TTS	**Sect 3**	16926	16669	7232	8179	9281	10546	11969	13465	14586	15097	15529
	Network	40128	39875	30574	31522	32623	33888	35288	36749	37842	38339	38760

After the calibration of $\hat{\rho}_2$ and $\hat{\rho}_3$, $K_{R,2}$ and $K_{R,3}$ will be calibrated by the same method. Tables B.11 and B.12 show the calibration results where $K_{R,2}$ = 0.3 and $K_{R,3}$ = 0.1 are optimal.

Table B.11: Calibration of $K_{R,2}$ for ALINEA-C (demand 3)

$K_{R,2}$	0.10	0.20	0.25	0.30	0.35	0.40	0.50	1.00	1.50	2.00	2.30
Network TTS	36082	30590	30581	30574	30574	30575	30579	30583	31134	34443	35679

Table B.12: Calibration of $K_{R,3}$ for ALINEA-C (demand 3)

$K_{R,3}$	0.10	0.20	0.30	0.40	0.50	1.00	1.50	2.00	2.30
Network TTS	30541	30567	30574	30576	30575	30571	30573	39008	39582

ALINEA-D

Calibration results of ALINEA-D are shown in table B.13 to B.16, from where the optimal parameters are selected as: $\hat{\rho}_2$ = 0.96 ρ_{crit}, $\hat{\rho}_3$ = ρ_{crit}, $K_{R,2}$ = 0.2 and $K_{R,3}$ = 0.1.

Table B.13 Calibration of $\hat{\rho}_2$ for ALINEA-D (demand 3)

$\hat{\rho}_2$	ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.9 ρ_{crit}
TTS of Sect 2	17663	17663	16851	16872	9668	9941	12333	12673	14849	15324	15750

Table B.14 Calibration of $\hat{\rho}_3$ for ALINEA-D (demand 3)

$\hat{\rho}_3$		ρ_{crit}	0.99 ρ_{crit}	0.98 ρ_{crit}	0.97 ρ_{crit}	0.96 ρ_{crit}	0.95 ρ_{crit}	0.94 ρ_{crit}	0.93 ρ_{crit}	0.92 ρ_{crit}	0.91 ρ_{crit}	0.9 ρ_{crit}
TTS	**Sect 3**	7439	7469	7511	7539	7580	7611	7651	7680	7711	7751	9931
	Network	31781	31811	31853	31881	31922	31953	31993	32022	32053	32093	34272

Table B.15 Calibration of $K_{R,2}$ for ALINEA-D (demand 3)

$K_{R,2}$	0.10	0.15	0.20	0.25	0.30	0.40	0.50	1.00	1.50	2.00	2.30
Network TTS	30874	30814	30811	31126	31781	31837	31932	32885	32841	37778	38171

Table B.16 Calibration of $K_{R,3}$ for ALINEA-D (demand 3)

$K_{R,3}$	0.10	0.20	0.30	0.40	0.50	1.00	1.50	2.00	2.30
Network TTS	30811	38988	39452	39443	39427	39380	39360	39360	39360

APPENDIX C

COLLECTED TRAFFIC DATA

C.1 Example of JTDB Data

Link ID	Link Description	Date	Time Period	Day Category	Quality	Avg Travel Time (secs)	Avg Travel Speed (km/h)	Total Flow (vehicles)
LM1022	M6 J10 to M6 J9	01/05/2011	00:00 - 00:15	Sunday	High	81.71	106.18	233
LM1022	M6 J10 to M6 J9	01/05/2011	00:15 - 00:30	Sunday	High	82.9	104.66	228.5
LM1022	M6 J10 to M6 J9	01/05/2011	00:30 - 00:45	Sunday	High	82.55	105.1	210.5
LM1022	M6 J10 to M6 J9	01/05/2011	00:45 - 01:00	Sunday	High	81.35	106.65	191
LM1022	M6 J10 to M6 J9	01/05/2011	01:00 - 01:15	Sunday	High	80.9	107.25	189.5
LM1022	M6 J10 to M6 J9	01/05/2011	01:15 - 01:30	Sunday	High	85.81	101.11	168.5
LM1022	M6 J10 to M6 J9	01/05/2011	01:30 - 01:45	Sunday	High	83.61	103.77	153
LM1022	M6 J10 to M6 J9	01/05/2011	01:45 - 02:00	Sunday	High	82.44	105.24	144
LM1022	M6 J10 to M6 J9	01/05/2011	02:00 - 02:15	Sunday	High	81.73	106.15	115
LM1022	M6 J10 to M6 J9	01/05/2011	02:15 - 02:30	Sunday	High	84.5	102.68	103.5
LM1022	M6 J10 to M6 J9	01/05/2011	02:30 - 02:45	Sunday	High	84.19	103.05	107.5
LM1022	M6 J10 to M6 J9	01/05/2011	02:45 - 03:00	Sunday	High	81.17	106.89	100.5
LM1022	M6 J10 to M6 J9	01/05/2011	03:00 - 03:15	Sunday	High	84.6	102.55	97.5
LM1022	M6 J10 to M6 J9	01/05/2011	03:15 - 03:30	Sunday	High	84.52	102.65	100
LM1022	M6 J10 to M6 J9	01/05/2011	03:30 - 03:45	Sunday	High	86.6	100.18	115.5
LM1022	M6 J10 to M6 J9	01/05/2011	03:45 - 04:00	Sunday	High	85.34	101.66	102
LM1022	M6 J10 to M6 J9	01/05/2011	04:00 - 04:15	Sunday	High	83.98	103.31	109.5
LM1022	M6 J10 to M6 J9	01/05/2011	04:15 - 04:30	Sunday	High	84.76	102.36	109.5

LM1022	M6 J10 to M6 J9	01/05/2011	04:30 - 04:45	Sunday	High	87.16	99.54	114
LM1022	M6 J10 to M6 J9	01/05/2011	04:45 - 05:00	Sunday	High	85.56	101.4	99
LM1022	M6 J10 to M6 J9	01/05/2011	05:00 - 05:15	Sunday	High	84.24	102.99	118
LM1022	M6 J10 to M6 J9	01/05/2011	05:15 - 05:30	Sunday	High	83.5	103.9	133.5
LM1022	M6 J10 to M6 J9	01/05/2011	05:30 - 05:45	Sunday	High	83.2	104.28	142
LM1022	M6 J10 to M6 J9	01/05/2011	05:45 - 06:00	Sunday	High	84.45	102.74	144.5
LM1022	M6 J10 to M6 J9	01/05/2011	06:00 - 06:15	Sunday	High	82.88	104.68	158
LM1022	M6 J10 to M6 J9	01/05/2011	06:15 - 06:30	Sunday	High	82.75	104.85	192
LM1022	M6 J10 to M6 J9	01/05/2011	06:30 - 06:45	Sunday	High	81.54	106.4	219.5
LM1022	M6 J10 to M6 J9	01/05/2011	06:45 - 07:00	Sunday	High	83.05	104.47	212
LM1022	M6 J10 to M6 J9	01/05/2011	07:00 - 07:15	Sunday	High	81.77	106.1	219.5
LM1022	M6 J10 to M6 J9	01/05/2011	07:15 - 07:30	Sunday	High	82.66	104.96	264
LM1022	M6 J10 to M6 J9	01/05/2011	07:30 - 07:45	Sunday	High	82.31	105.4	312.5
LM1022	M6 J10 to M6 J9	01/05/2011	07:45 - 08:00	Sunday	High	83.28	104.18	344.5
LM1022	M6 J10 to M6 J9	01/05/2011	08:00 - 08:15	Sunday	High	82.22	105.52	366
LM1022	M6 J10 to M6 J9	01/05/2011	08:15 - 08:30	Sunday	High	82.47	105.2	423
LM1022	M6 J10 to M6 J9	01/05/2011	08:30 - 08:45	Sunday	High	82.27	105.46	473
LM1022	M6 J10 to M6 J9	01/05/2011	08:45 - 09:00	Sunday	High	82.17	105.59	533.5
LM1022	M6 J10 to M6 J9	01/05/2011	09:00 - 09:15	Sunday	High	82.97	104.57	592.5
LM1022	M6 J10 to M6 J9	01/05/2011	09:15 - 09:30	Sunday	High	83.24	104.23	688.5
LM1022	M6 J10 to M6 J9	01/05/2011	09:30 - 09:45	Sunday	High	84.93	102.15	807
LM1022	M6 J10 to M6 J9	01/05/2011	09:45 - 10:00	Sunday	High	84.41	102.78	906
LM1022	M6 J10 to M6 J9	01/05/2011	10:00 - 10:15	Sunday	High	85.33	101.68	881
LM1022	M6 J10 to M6 J9	01/05/2011	10:15 - 10:30	Sunday	High	86.53	100.27	1037.5
LM10	M6 J10 to	01/05/2011	10:30 -	Sunday	High	92.47	93.82	1058

22	M6 J9		10:45					
LM1022	M6 J10 to M6 J9	01/05/2011	10:45 - 11:00	Sunday	High	95.96	90.41	1139
LM1022	M6 J10 to M6 J9	01/05/2011	11:00 - 11:15	Sunday	High	96.21	90.18	1158
LM1022	M6 J10 to M6 J9	01/05/2011	11:15 - 11:30	Sunday	High	95.38	90.96	1176.5
LM1022	M6 J10 to M6 J9	01/05/2011	11:30 - 11:45	Sunday	High	97.4	89.08	1190
LM1022	M6 J10 to M6 J9	01/05/2011	11:45 - 12:00	Sunday	High	96.71	89.71	1193.5
LM1022	M6 J10 to M6 J9	01/05/2011	12:00 - 12:15	Sunday	High	96.69	89.73	1219
LM1022	M6 J10 to M6 J9	01/05/2011	12:15 - 12:30	Sunday	High	96.37	90.03	1245
LM1022	M6 J10 to M6 J9	01/05/2011	12:30 - 12:45	Sunday	High	96.34	90.06	1231.5
LM1022	M6 J10 to M6 J9	01/05/2011	12:45 - 13:00	Sunday	High	97.3	89.17	1225
LM1022	M6 J10 to M6 J9	01/05/2011	13:00 - 13:15	Sunday	High	98.68	87.92	1185.5
LM1022	M6 J10 to M6 J9	01/05/2011	13:15 - 13:30	Sunday	High	99.69	87.03	1189.5
LM1022	M6 J10 to M6 J9	01/05/2011	13:30 - 13:45	Sunday	High	96.68	89.74	1193
LM1022	M6 J10 to M6 J9	01/05/2011	13:45 - 14:00	Sunday	High	95.71	90.65	1173.5
LM1022	M6 J10 to M6 J9	01/05/2011	14:00 - 14:15	Sunday	High	94.67	91.64	1170.5
LM1022	M6 J10 to M6 J9	01/05/2011	14:15 - 14:30	Sunday	High	95.5	90.85	1149.5
LM1022	M6 J10 to M6 J9	01/05/2011	14:30 - 14:45	Sunday	High	95.51	90.84	1161.5
LM1022	M6 J10 to M6 J9	01/05/2011	14:45 - 15:00	Sunday	High	94.13	92.17	1169
LM1022	M6 J10 to M6 J9	01/05/2011	15:00 - 15:15	Sunday	High	94.46	91.85	1178
LM1022	M6 J10 to M6 J9	01/05/2011	15:15 - 15:30	Sunday	High	94.72	91.6	1169.5
LM1022	M6 J10 to M6 J9	01/05/2011	15:30 - 15:45	Sunday	High	94.92	91.4	1110
LM1022	M6 J10 to M6 J9	01/05/2011	15:45 - 16:00	Sunday	High	95.14	91.19	1135
LM1022	M6 J10 to M6 J9	01/05/2011	16:00 - 16:15	Sunday	High	94.64	91.67	1145.5
LM1022	M6 J10 to M6 J9	01/05/2011	16:15 - 16:30	Sunday	High	95.36	90.98	1133.5
LM1022	M6 J10 to M6 J9	01/05/2011	16:30 - 16:45	Sunday	High	95.78	90.58	1129

LM1022	M6 J10 to M6 J9	01/05/2011	16:45 - 17:00	Sunday	High	103.99	83.43	1182.5
LM1022	M6 J10 to M6 J9	01/05/2011	17:00 - 17:15	Sunday	High	101	85.9	1171
LM1022	M6 J10 to M6 J9	01/05/2011	17:15 - 17:30	Sunday	High	96.67	89.75	1160.5
LM1022	M6 J10 to M6 J9	01/05/2011	17:30 - 17:45	Sunday	High	96.85	89.58	1175
LM1022	M6 J10 to M6 J9	01/05/2011	17:45 - 18:00	Sunday	High	95.58	90.77	1060
LM1022	M6 J10 to M6 J9	01/05/2011	18:00 - 18:15	Sunday	High	96.33	90.07	1081
LM1022	M6 J10 to M6 J9	01/05/2011	18:15 - 18:30	Sunday	High	95.51	90.84	1110
LM1022	M6 J10 to M6 J9	01/05/2011	18:30 - 18:45	Sunday	High	94.53	91.78	1068
LM1022	M6 J10 to M6 J9	01/05/2011	18:45 - 19:00	Sunday	High	94.78	91.54	1011.5
LM1022	M6 J10 to M6 J9	01/05/2011	19:00 - 19:15	Sunday	High	94.03	92.27	902.5
LM1022	M6 J10 to M6 J9	01/05/2011	19:15 - 19:30	Sunday	High	93.28	93.01	962
LM1022	M6 J10 to M6 J9	01/05/2011	19:30 - 19:45	Sunday	High	84.95	102.13	898
LM1022	M6 J10 to M6 J9	01/05/2011	19:45 - 20:00	Sunday	High	85.04	102.02	860
LM1022	M6 J10 to M6 J9	01/05/2011	20:00 - 20:15	Sunday	High	84.27	102.95	846.5
LM1022	M6 J10 to M6 J9	01/05/2011	20:15 - 20:30	Sunday	High	83.14	104.36	835.5
LM1022	M6 J10 to M6 J9	01/05/2011	20:30 - 20:45	Sunday	High	82.82	104.76	785.5
LM1022	M6 J10 to M6 J9	01/05/2011	20:45 - 21:00	Sunday	High	82.62	105.01	701
LM1022	M6 J10 to M6 J9	01/05/2011	21:00 - 21:15	Sunday	High	82.98	104.56	687
LM1022	M6 J10 to M6 J9	01/05/2011	21:15 - 21:30	Sunday	High	82.12	105.65	611.5
LM1022	M6 J10 to M6 J9	01/05/2011	21:30 - 21:45	Sunday	High	82.46	105.21	587.5
LM1022	M6 J10 to M6 J9	01/05/2011	21:45 - 22:00	Sunday	High	81.69	106.21	557.5
LM1022	M6 J10 to M6 J9	01/05/2011	22:00 - 22:15	Sunday	High	82.31	105.41	451
LM1022	M6 J10 to M6 J9	01/05/2011	22:15 - 22:30	Sunday	High	82.37	105.33	459.5
LM1022	M6 J10 to M6 J9	01/05/2011	22:30 - 22:45	Sunday	High	82.87	104.69	395.5
LM10	M6 J10 to	01/05/2011	22:45 -	Sunday	High	82.46	105.21	337

22	M6 J9		23:00					
LM1022	M6 J10 to M6 J9	01/05/2011	23:00 - 23:15	Sunday	High	82.38	105.32	319
LM1022	M6 J10 to M6 J9	01/05/2011	23:15 - 23:30	Sunday	High	82.34	105.37	322.5
LM1022	M6 J10 to M6 J9	01/05/2011	23:30 - 23:45	Sunday	High	81.27	106.76	273
LM1022	M6 J10 to M6 J9	01/05/2011	23:45 - 00:00	Sunday	High	82.42	105.26	225

C.2 Example of TRADS Data

Period report between Sunday 1st May 2011 and Monday 1st Aug 2011 for site 9/30030314, , M6, MIDAS site at M6/5993B, 050/6/024/321 on M6 southbound between J10A and J10(E399061, N299818) view site location on map

15 Min Flows

	b Mon x11	Tue x12	Wed x13	Thu x13	Fri x13	Sat x11	Sun x11	Mn-Fr Mean	Mn-Sn Mean
00:15	230	207	219	227	216	235	220	219	222
00:30	212	185	208	205	207	239	211	203	209
00:45	189	165	185	178	194	219	194	182	189
01:00	164	151	166	162	172	203	196	163	173
01:15	158	138	157	160	153	189	168	153	160
01:30	139	131	144	162	150	184	158	145	152
01:45	129	123	138	139	138	168	142	133	139
02:00	120	115	137	132	142	163	129	129	134
02:15	122	125	140	136	141	156	113	132	133
02:30	112	130	142	146	154	160	112	136	136
02:45	108	137	143	147	149	154	94	136	133
03:00	119	132	134	141	151	146	91	135	130
03:15	126	149	156	165	174	161	88	154	145
03:30	148	174	178	170	182	162	103	170	159
03:45	180	190	201	194	199	162	108	192	176
04:00	201	199	205	199	200	164	107	200	182
04:15	218	216	221	220	225	184	107	220	198
04:30	272	237	234	231	226	191	106	240	213
04:45	343	283	278	283	272	192	116	291	252
05:00	405	346	321	322	304	214	114	339	289
05:15	493	420	393	401	372	244	119	415	348
05:30	606	527	502	505	460	297	137	520	433
05:45	735	649	630	610	552	336	156	635	524
06:00	805	740	703	691	619	356	164	711	582
06:15	940	907	891	843	787	403	181	873	707
06:30	1119	1173	1147	1132	1031	492	236	1120	904
06:45	1182	1295	1324	1289	1175	562	271	1253	1014
07:00	985	1265	1253	1259	1162	568	278	1184	967
07:15	862	1085	1189	1212	1101	573	289	1089	901
07:30	844	1074	1130	1128	1123	656	336	1059	898
07:45	923	1044	1100	1089	1110	709	376	1053	907
08:00	948	958	1087	1114	1066	738	401	1034	901

08:15	918	921	1069	1049	1022	735	414	995	875
08:30	973	897	948	1039	995	810	471	970	876
08:45	958	988	920	990	953	875	512	961	885
09:00	963	1027	949	970	954	868	560	972	898
09:15	909	977	1001	1031	940	918	616	971	913
09:30	932	990	984	1031	972	985	704	981	942
09:45	956	962	976	1079	1010	1065	777	996	975
10:00	986	998	947	1062	1051	1109	880	1008	1004
10:15	1017	932	962	1091	1078	1118	916	1016	1016
10:30	1020	939	987	1077	1095	1139	991	1023	1035
10:45	1095	1012	992	1083	1178	1185	1040	1072	1083
11:00	1100	973	989	1114	1193	1177	1104	1073	1092
11:15	1123	1006	1032	1101	1208	1159	1114	1094	1106
11:30	1133	985	1036	1104	1169	1175	1159	1085	1108
11:45	1198	1037	1045	1084	1240	1171	1203	1120	1139
12:00	1174	1036	1081	1095	1190	1158	1181	1115	1130
12:15	1162	1052	1047	1095	1200	1118	1134	1111	1115
12:30	1143	1071	1080	1100	1179	1113	1169	1114	1122
12:45	1170	1031	1073	1106	1187	1090	1152	1113	1115
13:00	1132	1059	1036	1106	1199	1090	1188	1106	1115
13:15	1090	1065	1066	1065	1194	1046	1202	1096	1104
13:30	1113	1053	1065	1111	1160	1092	1148	1100	1106
13:45	1097	1036	1110	1088	1137	1066	1162	1093	1099
14:00	1088	991	1070	1112	1108	1053	1128	1073	1078
14:15	1081	1004	1071	1100	1081	1019	1068	1067	1060
14:30	1090	1062	1121	1126	1208	981	1101	1121	1098
14:45	1089	1038	1103	1157	1198	958	1109	1117	1093
15:00	1084	1064	1106	1142	1194	962	1107	1118	1094
15:15	1111	1075	1079	1122	1145	935	1100	1106	1081
15:30	1066	1094	1164	1141	1159	941	1073	1124	1091
15:45	1102	1090	1180	1201	1197	920	1127	1154	1116
16:00	1101	1150	1209	1213	1158	884	1109	1166	1117
16:15	1120	1177	1235	1232	1151	865	1117	1183	1128
16:30	1175	1210	1307	1258	1129	931	1134	1215	1163
16:45	1208	1223	1298	1274	1102	880	1153	1221	1162
17:00	1196	1188	1266	1218	1097	859	1132	1193	1136
17:15	1166	1192	1227	1246	1114	883	1136	1189	1137
17:30	1216	1152	1226	1244	1166	875	1109	1200	1141
17:45	1156	1157	1206	1196	1127	876	1114	1168	1118
18:00	1101	1102	1181	1119	1081	874	1148	1116	1086
18:15	1007	1062	1112	1074	1037	827	1127	1058	1035
18:30	979	1029	1049	1050	1031	829	1139	1027	1015
18:45	931	920	987	1053	1029	811	1119	984	978

19:00	844	819	938	938	1011	760	1089	910	914
19:15	837	813	901	900	994	730	1039	889	887
19:30	768	754	831	865	989	679	1058	841	849
19:45	738	691	766	821	944	644	1050	792	807
20:00	690	660	671	762	917	595	1043	740	762
20:15	656	621	642	714	883	575	991	703	726
20:30	658	604	629	697	854	560	959	688	708
20:45	600	557	570	630	776	502	889	626	646
21:00	562	518	524	576	680	461	837	572	594
21:15	511	485	481	534	618	418	788	525	547
21:30	489	450	459	516	566	405	759	496	520
21:45	447	412	429	462	526	374	685	455	476
22:00	418	385	395	420	448	355	587	413	429
22:15	377	358	364	393	406	328	504	379	390
22:30	359	337	348	362	389	320	452	359	366
22:45	322	300	318	329	349	328	403	323	335
23:00	291	281	309	298	322	287	363	300	307
23:15	260	269	296	290	296	271	319	282	285
23:30	244	266	274	289	287	264	286	272	272
23:45	240	259	277	281	284	260	272	268	267
24:00:00	216	245	255	240	260	240	248	243	243

Totals

07-19hr	50850	50007	52036	53430	53427	45861	46638	51950	50321
06-22hr	62450	61597	63949	65850	66777	54184	58289	64124	61870
06-24hr	64759	63912	66390	68332	69370	56482	61136	66552	64340
00-24hr	71093	69781	72325	74258	75122	61361	64389	72515	69761

b Bank Holiday *w* Weather *a* Accident *t* Time change *r* Road Works

Showing only complete days. No estimated data. Not including hidden data.