Models and Algorithms for the Optimization of Traffic Flows and Emissions Using Dynamic Routing and Pricing

by

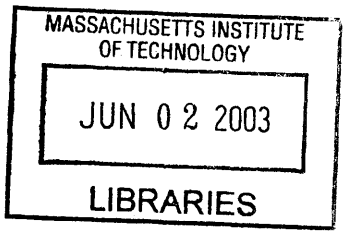
Maya Abou Zeid

Bachelor of Engineering in Civil and Environmental Engineering
American University of Beirut, 2001

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Transportation
at the
Massachusetts Institute of Technology

June 2003



Author .....
Department of Civil and Environmental Engineering
May 9, 2003

Certified by .....
Ismail Chabini
Associate Professor, Civil and Environmental Engineering
Thesis Supervisor

Accepted by .....
Oral Buyukozturk
Chairman, Departmental Committee on Graduate Studies

Models and Algorithms for the Optimization of Traffic Flows and Emissions Using Dynamic Routing and Pricing

by
Maya Abou Zeid

Submitted to the Department of Civil and Environmental Engineering
on May 9, 2003, in partial fulfillment of the requirements for the degree of
Master of Science in Transportation

Abstract

The research documented in this thesis is centered on the development and evaluation of models and algorithms for the optimization of traffic flows and emissions via routing and pricing in dynamic traffic networks. A set of problems that arise in this context are studied. These include: (1) the development of a probabilistic approach to model acceleration, (2) the study of the dynamic capacitated minimum cost flow problem, (3) an experimental analysis of improvements in shortest path algorithms, and (4) the study of dynamic congestion and emission pricing.

We propose a probabilistic approach for modeling accelerations and decelerations in traffic networks as random variables that are a function of speed and road type. We use the approach to integrate a non-microscopic dynamic traffic model and an instantaneous emission model.

We develop routing algorithms that can be used in the context of traffic flow optimization. First, we study the capacitated minimum cost flow problem in dynamic traffic networks, and develop two solution algorithms for the problem. The developed algorithms are shown experimentally to be more efficient than an existing algorithm in the literature. Second, we perform experimental testing to assess the computational performance of a new approach to solve the shortest path problem in static and dynamic FIFO networks, that tries to overcome some of the limitations in traditional comparison-based label-setting algorithms.

Finally, we develop a second-best link-based dynamic congestion pricing model and formulate it as a bi-level program. We develop solution algorithms based on sensitivity analysis, and model both route and departure time choices as users' reaction to the prices. We extend the model and algorithms to study emission pricing. Finally, we formulate the model with additional travel time or emissions constraints, and evaluate the effectiveness of the pricing methods on small hypothetical network examples.

Thesis Supervisor: Ismail Chabini
Title: Associate Professor, Civil and Environmental Engineering

Acknowledgements

Many thanks to a highly talented and energetic Professor Ismail Chabini, my research supervisor, for providing guidance and insights throughout all stages of my research. Thanks for continuously following up on my research, for patiently explaining to me a lot of concepts, and for giving me the opportunity to work on topics that I like.

I owe a lot of where I am today to Professor Isam Kaysi, my undergraduate advisor. I thank him for introducing me to the transportation field, for his confidence in me, and for all the invaluable advice and support that he offered to me.

Thanks to Ford Motor Company, MIT Presidential office, MIT Civil and Environmental Engineering Department, and the National Science Foundation for funding my research and studies.

Thanks to Dr. Edward Nam from the Ford Scientific Research Laboratory for helping me in several parts of my research, and for his friendship. Thanks to Alessandra Cappiello for taking the lead in our research project, for explaining to me all about her research and helping me in mine, and for being a great friend. Thanks also to my research colleagues in the ACTS group, especially to Hai Jiang. Thanks to Marwan Abou-Zeid for his help and advice. Thanks to Dr. David Chock for his wise technical counsel.

Thanks to my friends at MIT for cheering up my days: Alejandro & Tania, Ale & Alex, Miguel, Akhil, Michael, Phani, Ging Ging, Biova, Costas, Hai, Ramsay, Eva; to my Lebanese friends for always being there: Carol, Lara, Bassel, Fadi, Cesar & Laura, Christian, Asmahan & Ivan, Rayan, Walid, Elsa, Ziad, Elie, and Nisrine; and to my wonderful roomies: Lara, Anne, Elsa, and Erisa.

Thanks to my dear family George, Aida, and Marwan for their love.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis presents models and algorithms for the optimization of traffic flows and emissions in dynamic (time-dependent) traffic networks via routing and pricing. This chapter provides a motivation for and a context in which congestion and emissions are modeled and managed, and presents the objectives, contributions, and organization of the thesis.

1.1 Traffic Congestion and Emissions

Traffic congestion has been a problem facing urban commuters for several decades, but has been lately intensified by the growth in automobile ownership and changes in land-use and development patterns. The economic inefficiencies due to congestion, manifested in the form of extended travel delays, wasted fuel, and adjustments to activity patterns, have led communities worldwide to devise measures that aim at its mitigation.

Congestion relief measures have generally been classified into supply and demand measures. Supply measures are those that adapt the network capacity to the existing demand, by investing in infrastructure (such as building more roads), enhancing public transport services, or improving the utilization of the existing network (such as by traffic signal control, ramp metering, driver information systems, etc.). Demand measures, on the other hand, are those that adapt the demand to the existing road capacity by influencing users' travel behavior. Examples of these strategies include using fiscal measures (such as road pricing, parking pricing, fuel taxes, etc.), adopting alternative work schedules (staggered

shifts, flextime policies, compressed workweek), encouraging ridesharing, and telecommuting.

Although congestion relief strategies might vary in their short-term effectiveness, their long-term consequences should be carefully analyzed. It has been argued that urban highways have a tendency to achieve peak congestion levels irrespective of supply and demand (Downs (1992), Small (1992 a)). Most congestion relief strategies might reduce the duration of peak-hour traffic congestion but not its intensity as commuters would change their routes, trip times, and modes of travel to occupy any road space freed by such strategies – a phenomenon referred to in Downs (1992) as the principle of triple convergence[1].

Besides congestion, road transportation is a major source of mobile-source emissions. It is estimated that motor vehicles in the Unites States are responsible for about 90 % of carbon monoxide in the air and 50 % of smog and hazardous air pollutants (EPA's website: www.epa.gov). The accumulation of emissions leads to the concentration of pollutants in the atmosphere, which are also affected by meteorology, topography, and further chemical reactions. Vehicular emissions and pollutant concentrations are governed by emission and air quality standards, set by regulatory agencies such as the U. S. Environmental Protection Agency. Vehicle emission standards define the maximum allowable tailpipe emissions given a vehicle's age and mileage. Air quality standards set upper limits on the concentrations of pollutants in the atmosphere, using various temporal aggregations. In order to achieve environmental objectives and/or comply with environmental standards, there is need to implement adequate policies.

In general, reductions in congestion are believed to be positively correlated with reductions in emissions and subsequent improvements in air quality. One could then apply the methods previously described for congestion relief to reduce emissions. It should be noted though that in certain instances, paradoxes could exist between traffic-improving measures and subsequent emission levels (see for instance Nagurney (2000 a)). For

[1] Triple convergence is defined in Downs (1992) as a combination of (1) spatial convergence, where travelers formerly using congested roads switch to the faster route on which improvements have been made, (2) time convergence, where travelers previously traveling before or after the congested period switch their departure times to the peak period, and (3) modal convergence, where some travelers previously using public transit now switch to using the private auto as driving has become faster. The result of the three types of convergence is that traffic conditions prior to the improvement would continue to prevail after the improvement.

example, in Liu (2003) the emission level consequences of ramp metering, signal coordination, and demand management are investigated. It is demonstrated that although at a network level total emissions might decrease, this need not be the case at a local level (such as in the proximity of the traffic signals). Other policies that specifically aim at reducing emissions include vehicle technology measures (such as using cleaner fuels or more effective catalytic converters), inspection and maintenance programs, market-based incentives such as emission pricing, etc.

Consequently, traffic management strategies should be assessed through multi-criteria analysis methods that consider both congestion and emissions, and possibly other criteria such as safety and equity. In order to support the development and evaluation of such strategies, mathematical models of traffic flows, emissions, and dispersion are needed. Traffic models simulate the reaction of users to a given management method and, together with emission and air quality models, send back indicators of congestion, emissions, and air quality. A traffic-emissions-air quality laboratory consists of the modules shown in Figure 1.1.

Traffic models determine an equilibrium between supply and demand given a network topology, a matrix of origin-destination demands, user behavioral models (e.g. route and departure time choices), and link performance functions. They can be classified as static or dynamic (time-dependent). The reader interested in a review of traffic models is referred to Cascetta (2001) and Hoogendoorn and Bovy (2001). The outputs from a traffic model, namely vehicle speeds and accelerations (if available), are fed into an emission model. Emission models can also be classified as static average speed-based models or dynamic models that take both speed and acceleration as input. An overview of emission models is provided in Cappiello (2002). Given a vehicle's technology specification, operating conditions (speed and acceleration), and external environmental conditions, an emission model outputs the amounts of emissions (by emission species) generated. These constitute the input to dispersion and photochemical models, which predict further reactions of the emitted species, model their dispersion in the air, and eventually output pollutant concentrations as indicators of air quality. Readers interested in a review of dispersion and air quality models are referred to Barratt (2001).

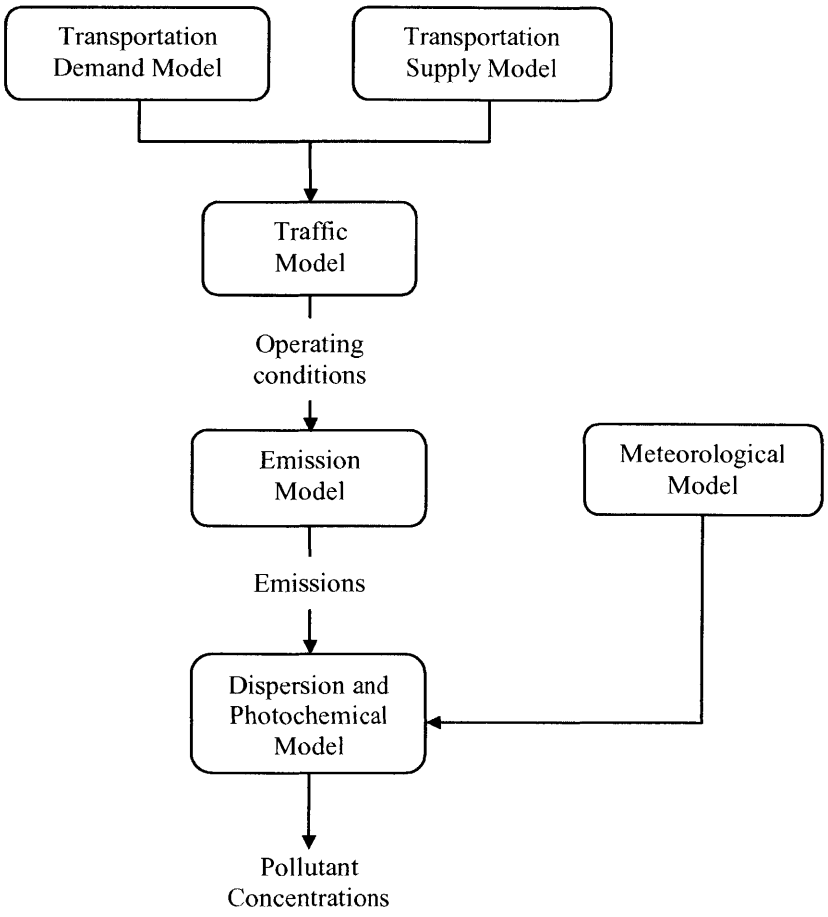


Figure 1.1. Components of a traffic-emissions-air quality simulation laboratory. (from Cappiello (2002))

Considerable attention from the research community has been devoted for enhancing the prediction accuracy of traffic models and emission models separately. Recently, there has been an interest in the integration of these models at various levels of detail, a review of which is provided in Cappiello (2002) and Liu (2003). For instance, a microscopic traffic simulator can easily be integrated with an instantaneous emission model as the former generates both speed and acceleration as input to the latter. The integration of a non-microscopic traffic model, which generates speeds but not accelerations as output, with an instantaneous emission model is not as straightforward. We describe later in this thesis a method that aids in this integration.

In addition to the modeling requirements of traffic management strategies, there is need to develop algorithms that solve the models. Although the development of algorithms is

usually motivated by the need to solve a particular problem, the algorithms can often be used in a variety of application settings. Moreover, if the algorithms are used in a real-time context, such as in Intelligent Transportation System applications, they should be efficient to meet the running time requirements of the sub-problems in which they arise. In this thesis, the algorithms that we develop will be aimed at solving dynamic routing and pricing problems for the optimization of travel times and/or emissions in dynamic traffic networks.

1.2 Thesis Objectives and Contributions

The research documented in this thesis has the following main directions: (1) the development of methods that enable the enhanced representation of traffic flows and emissions and that aid in the integration of non-microscopic dynamic traffic models and instantaneous emission models, (2) the development of models and management algorithms for congestion and emissions in dynamic traffic networks, (3) the implementation of the developed models and algorithms in a simulation laboratory, and (4) the assessment of the effectiveness of the proposed methods by testing them on hypothetical network examples.

More specifically, the main contributions of this thesis with respect to enhancing the modeling requirements of traffic flows and emissions are: (1) the development of a probabilistic approach to model accelerations as a function of speed and road type in traffic networks, with applications in the integration of instantaneous emission models and non-microscopic traffic models, and (2) the addition of a departure time choice model to an existing analytical dynamic traffic assignment model.

In the area of management models and algorithms, the main contributions are: (1) the study of the minimum cost flow problem in capacitated time-dependent networks and the development of efficient solution algorithms for this problem, (2) the experimental study of a new approach for solving the shortest path problem in static and dynamic First-In-First-Out (FIFO) networks that is different from traditional comparison-based label-setting algorithms, and (3) the formulation of a second-best link-based model of congestion and emission pricing in dynamic traffic networks as a bi-level program and the development of solution algorithms for this problem.

1.3 Thesis Organization

This thesis is organized as follows. Chapter 2 provides a methodology for modeling accelerations in traffic networks as random variables that are functions of speed and road type. The probabilistic approach is applied to a data set, for which an acceleration and a deceleration distribution is developed for every speed range and road type. The methodology, results, and applications of the acceleration model are presented. Results from this research have been reported in Abou Zeid et al. (2002).

Chapter 3 provides an in-depth treatment of the minimum cost flow problem in capacitated time-dependent networks, where a given amount of flow should be sent from a source node to a sink node at a certain departure time subject to link capacities. Solution algorithms are developed and computational tests are performed to assess the efficiency of the developed algorithms. The case of the minimum travel time flow problem is also addressed. The discussion is further extended to allow for waiting at nodes and for multiple origins, destinations, and departure times. Results from this research have been reported in Chabini and Abou Zeid (2003).

Chapter 4 presents a new approach developed in Chabini (2002) for solving the shortest path problem in static and dynamic FIFO networks, and provides experimental testing of the algorithms. The approach is based on defining optimality conditions for detecting whether a label is optimal, and utilizing these conditions to reduce the work needed for sorting labels, which is a bottleneck operation in traditional comparison-based label-setting algorithms.

Chapter 5 develops methods for congestion and emission pricing in dynamic traffic networks. A second-best link-based dynamic congestion pricing model is formulated as a bi-level program with the objective of minimizing total travel time subject to upper bounds on the link prices and to the users' reaction to the implemented prices. Solution algorithms are developed, and both route and departure time choices are modeled in the process of finding uses' reaction (equilibrium). The methodology is extended to study dynamic emission pricing in general traffic networks, where the prices vary additionally by vehicle category. Finally, both congestion and emissions are accounted for in the optimization process by adding total emissions (total travel time) constraints and hot spot environmental constraints

to the basic congestion (emission) pricing model. Several experiments are conducted to assess the effectiveness of the proposed pricing methods.

Chapter 6 concludes the thesis and gives directions for future research.

This thesis addresses a set of problems in the context of traffic flows and emissions optimization. These problems are studied separately in each of the chapters, and can be read independently of each other.

Chapter 2

Probabilistic Modeling of Acceleration in Traffic Networks as a Function of Speed and Road Type

2.1 Introduction

Characterizing travel behavior and vehicle activity has been an important research topic that has numerous applications in traffic and emission modeling. Current travel demand models give average speeds as outputs, which are not sufficient for the increasing data needs of emission modelers. Recent emission research recognizes that the engine mode of operation will be a significant variable in new modal emission models (Barth (1998), Guensler et al. (1998), Washington et al. (1998), Roberts et al. (1999)). When modal activity exceeds specific thresholds of variables such as power, positive kinetic energy, acceleration, or idle mode, emission levels can rise significantly. Thus, there is a need to understand how driving behavior and dynamic vehicular activity affect the proportion of driving spent in different modes (idling, cruising, acceleration, deceleration, etc.). Sierra Research (Carlson and Austin (1997)) developed representative driving cycles for different facility types and congestion levels after analyzing instrumented and chase car data. While this research was a

valuable addition to the literature, it does not present a statistical methodology for generating speed and acceleration distributions, which may be necessary for applications where specific driving cycles are an insufficient tool. Moreover, driver interaction with the vehicle in terms of acceleration and (to a lesser extent) deceleration patterns is important for emission modeling. For example, high-speed, high-acceleration, and heavy braking activities are typically exhibited by young male drivers, and this in turn increases emissions, while older drivers might drive more conservatively than younger drivers (Wolf et al. (1999), Fancher et al. (1998)). Driving patterns might also be dependent on the particular city and the nature of its transportation network, i.e. whether it is mostly dominated by local streets or freeways (LeBlanc et al. (1995)), and on the traffic and control conditions (Special Report: Highway Capacity Manual (1998)). For these reasons, it is important to study those aspects of driving behavior and road characteristics that may influence the statistics of acceleration and deceleration events for a given speed.

We develop a novel approach for the quantification of acceleration and deceleration events in a traffic network. The approach models acceleration as a random variable whose distribution varies as a function of speed and road type. The basic motivation of this research is to integrate non-microscopic dynamic traffic models and instantaneous emission models, though there are other applications as well. Non-microscopic dynamic traffic assignment models are fast, applicable on a regional scale, and generally easier to calibrate than their microscopic counterparts. Their basic limitation is that they describe network conditions in terms of average link speeds, but do not provide accelerations. Load-based emission models, however, require both speed and acceleration as input. Therefore, a probabilistic acceleration model is an efficient method of overcoming the shortcomings of non-microscopic traffic models and providing the necessary link to emission models. Even in the absence of a traffic model, the acceleration model is also useful when only speed data are available in a given city from field measurements. This leads, for instance, to more accurate emission modeling using instantaneous emission models rather than average speed-based emission models (such as EPA's MOBILE6), which in principle do not properly quantify emissions from vehicles under dynamic conditions (National Research Council (2000)) and are thus an approximation at best.

Several approaches for the quantification of acceleration and deceleration events can be found in the literature. TRANSIMS, a traffic simulator based on cellular automata modeling, uses aggregate real-world frequencies of power factor (2*speed*acceleration) to model the accelerations. The power factor is then used to estimate emissions (Williams et al. (1999)). Microscopic traffic models assign accelerations to individual vehicles based on the interaction among vehicles and the traffic regimes. Examples of these regimes are car-following and free-flowing. Therefore, in these models, acceleration is a function of parameters such as headway distribution, the relative difference in speed between adjacent vehicles, and driver reaction time (Ahmed (1999)).

In this research, however, the focus is to develop a probabilistic approach to estimate acceleration and deceleration activity from the mere knowledge of speed, without modeling vehicular interactions at the microscopic level. The analysis is conducted using data for four main road types: interstate highways, state highways, arterials, and collectors. It is well known that the potential for accelerating or decelerating decreases as speed increases because of power and traction limitations of the vehicle. However, these limitations are usually insufficient to describe the dynamics of vehicles. It is also necessary to determine the statistical distribution of accelerations and decelerations within a given speed range on a link, which is defined by the state variables of density and flow (or average speed).

This chapter is organized as follows. In Section 2.2, we describe the data for which we develop an acceleration model. In section 2.3, we present the approach that we developed to calibrate the model. In Section 2.4, we provide an analysis of the results. In Section 2.5, we describe an application of the model. Finally, in Section 2.6, we conclude the chapter and give future research directions.

2.2 Data

The data sets of this research are obtained in conjunction with an intelligent cruise control study sponsored by the U.S. Department of Transportation and conducted in South Eastern Michigan from July 1996 to September 1997 by the University of Michigan Transportation Research Institute (Fancher et al. (1998)). The developed model is built on real-world driving data collected during the first week of the study, during which the intelligent cruise

control was not functional. 108 randomly-chosen drivers from eight counties of South Eastern Michigan were selected to drive in metropolitan and rural areas of the state, using the same type of vehicle: an instrumented 1996 Chrysler Concorde.

Drivers were classified into five categories according to their driving behavior: (1) ultraconservative: means an unusual tendency towards far ('far' means large gap between leading and following vehicle) and/or slow driving, (2) planner: means an unusual tendency towards far and/or fast driving, (3) hunter/tailgater: means an unusual tendency towards fast and/or close driving, (4) extremist: means that the driver satisfies more than one of the above tendencies, and (5) flow conformist: means that the driver satisfies none of the above tendencies. A flow conformist tends to travel at the same speed as other cars and at approximately the median headway time-gap.

A sub-sample of eighteen drivers, each conducting 20 to 60 trips, was used to develop the model presented in this chapter. Most trip durations were less than 30 minutes. The eighteen drivers belong to the following categories: two planners, three extremists, five hunters, four ultraconservatives, and four flow conformists. The model does not capture differences in driver aggressiveness because the intent is to isolate road type as the only independent variable.

Roads were classified into the following types: high-speed ramp, interstate highway, state highway, arterial, collector, light duty, alley or unpaved, unknown, and low-speed ramp. Only four road types (interstate highways, state highways, arterials, and collectors) are considered in the present study because of data availability. Moreover, these road types cover most road types in a transportation network (except for on-ramps and off-ramps).

The distribution of acceleration and deceleration data points, aggregated from all drivers on every road type, is shown in Table 2.1. These observations correspond to second-by-second combinations of speed and acceleration.

Table 2.1. Number of observations of acceleration and deceleration by road type.

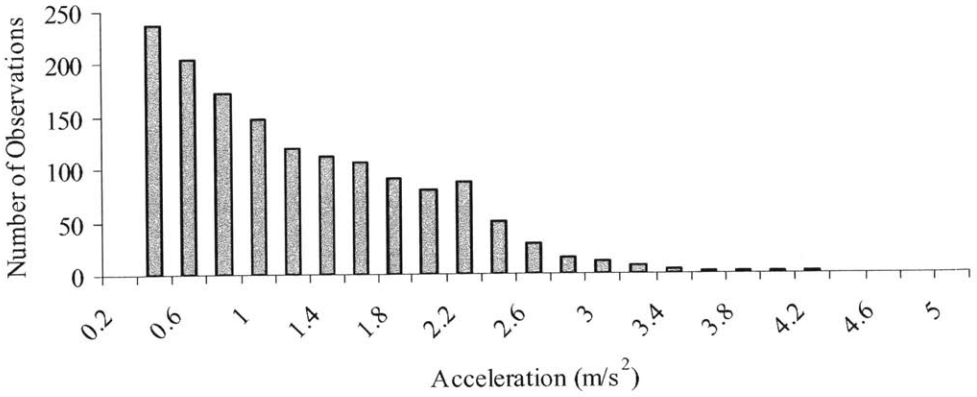
Road Type	Interstate Highway	State Highway	Arterial	Collector
Acceleration	5,597	3,633	20,580	5,804
Deceleration	12,569	6,074	31,586	11,043

2.3 Calibration

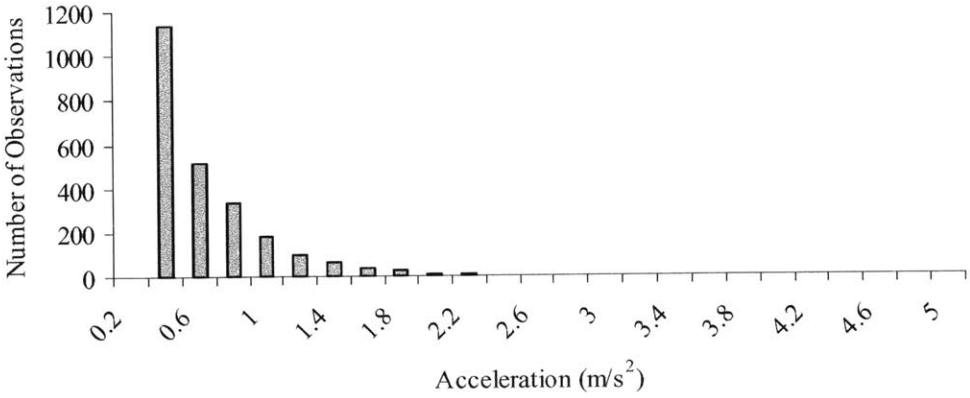
In this section, we describe the approach that we followed to develop an acceleration model corresponding to the data described above. The same procedure can be used in general to derive statistical acceleration distributions from any given acceleration data set though the fitted distributions might vary from one data set to another.

The first five trips have been eliminated from every driver's record to remove some of the "novelty factor" bias that might be present at the beginning of the test since drivers might not be accustomed to their new vehicles. The remaining data are divided into four subsets, one for each considered road type, to see whether acceleration and deceleration distributions are dependent on road type. The data in each subset are further divided into two groups: acceleration data (strictly positive values) and deceleration data (zero or negative values). Acceleration and deceleration are considered separately since in general they may not be similarly distributed. In the case of arterials, we have divided the data into two subsets, S_C for calibration and S_V for validation. For the other road types, we did not validate the model because of the lack of a sufficient number of observations for those road types.

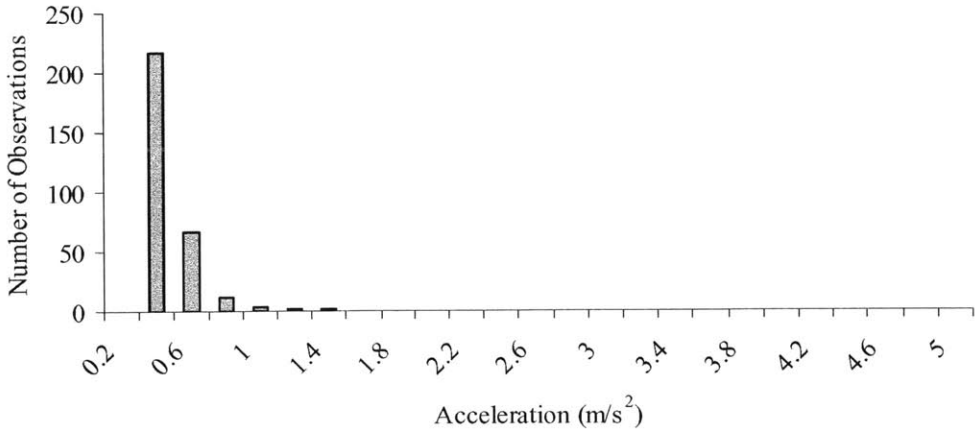
Consider one data group, for example, accelerations (strictly positive values) on road type r. The statistical distribution analysis is conducted for 10 km/h speed ranges. For each speed range v, we would like to find a probability distribution that fits the acceleration distribution obtained from the sample for that particular speed range. Plotting the sample acceleration values suggests a distribution similar (with a scale factor) to the density of a half-normal distribution with zero mean $(\mu = 0)$, and a standard deviation to be determined (see Figure 2.1).


Number of Observations
250
200
150
100
50
0
0.2
0.6
1
1.4
1.8
2.2
2.6
3
3.4
3.8
4.2
4.6
5
Acceleration (m/s^2)

(a)


Number of Observations
1200
1000
800
600
400
200
0
0.2
0.6
1
1.4
1.8
2.2
2.6
3
3.4
3.8
4.2
4.6
5
Acceleration (m/s^2)

(b)



(c)

Figure 2.1. Acceleration distribution for the calibration data set on arterials for the speed ranges 21-30 km/h (part a), 51-60 km/h (part b), and 81-90 km/h (part c).

Let $N^+_{r,v}$ be the total number of acceleration observations aggregated from all drivers driving on a road type r in a speed range v. Let $Q^+_{r,v}(a)$ be the sample probability of observing a certain value of acceleration a, where a belongs to an acceleration interval of length 0.2 m/s^2 $(e.g.(0.2,0.4],(0.4,0.6],...)$. If a belongs to an acceleration interval $[a_1,a_2]$, we define $T^+_{r,v}(a)$ as the total number of acceleration observations that are in the interval $[a_1,a_2]$.Then $Q^+_{r,v}(a)$ is given by:

$$Q^+_{r,v}(a)=\frac{T^+_{r,v}(a)}{N^+_{r,v}}. \tag{2.1}$$

The sample standard deviation is given by the expression:

$$\sigma^+_{r,v}=\sqrt{\sum_{a\in A}(a-\mu)^2 Q^+_{r,v}(a)}=\sqrt{\sum_{a\in A}(a-0)^2 Q^+_{r,v}(a)}=\sqrt{\sum_{a\in A}a^2 Q^+_{r,v}(a)}, \tag{2.2}$$

where A consists of all the acceleration values in the data set, taken at 0.2 m/s^2 intervals.

The half-normal probability density function, fitted to the acceleration data, is given by:

$$f^+_{r,v}(a)=\frac{2}{\sqrt{2\pi}\sigma^+_{r,v}}\exp\left[-0.5\left(\frac{a-\mu}{\sigma^+_{r,v}}\right)^2\right]=\frac{2}{\sqrt{2\pi}\sigma^+_{r,v}}\exp\left[-0.5\left(\frac{a}{\sigma^+_{r,v}}\right)^2\right], \tag{2.3}$$

where $\sigma^+_{r,v}$ is the standard deviation obtained from the acceleration sample, as given by expression (2.2).

The half-normal distribution is truncated at some maximum acceleration value a^+_v, so as to avoid predicting unrealistic high accelerations. The truncated distribution is then normalized so that the area under the resulting density function is one. The resulting density function is given by:

$$f^+_{normalized,r,v} = \frac{f^+_{r,v}(a)}{\int_0^{a^+_v} f^+_{r,v}(a)da}. \tag{2.4}$$

An error term $\varepsilon^+_{r,v}$, defined as the sum of the squares of the difference between the cumulative sample probability $F^+_{1,r,v}(a)$ and the cumulative truncated and normalized half-normal distribution function $F^+_{2,r,v}(a)$ is used to test the goodness-of-fit of the obtained distribution:

$$\varepsilon^+_{r,v} = \sum_{a\in A}\left[F^+_{2,r,v}(a) - F^+_{1,r,v}(a)\right]^2. \tag{2.5}$$

Low values of the error term $\varepsilon^+_{r,v}$ indicate a good fit, while high values suggest that the proposed distribution does not explain well the variation in acceleration. This procedure was applied for every road type and speed range in the acceleration data.

The maximum acceleration values at which the distributions are truncated are approximately equal to the maximum experimental values of acceleration and deceleration for every speed range, and they are given by: $a^+_v = 5$ m/s^2 for speed ranges from 0-10 to 71-80 km/h, $a^+_v = 2.5$ m/s^2 for speed range 81-90 km/h, $a^+_v = 1$ m/s^2 for speed range 91-100 km/h, $a^+_v = 0.75$ m/s^2 for speed range 101-110 km/h, and $a^+_v = 0.5$ m/s^2 for speed range 111-120 km/h.

The steps described above to estimate accelerations are applicable to the deceleration data for every road type and speed range.

The density function $f^*_{r,v}(a)$ of the combined distribution of acceleration and deceleration corresponding to road type r and speed range v is obtained by weighing the truncated and normalized distribution with the probability of occurrence of acceleration and deceleration, respectively:

$$f^*_{r,v}(a) = \begin{cases} P^-_{r,v} \dfrac{f^-_{r,v}}{\int_{a^-_v}^{0} f^-_{r,v}(a)da} & \text{for } a \le 0 \\ P^+_{r,v} \dfrac{f^+_{r,v}}{\int_{0}^{a^+_v} f^+_{r,v}(a)da} & \text{for } a > 0, \end{cases} \tag{2.6}$$

where $P^-_{r,v}$ and $P^+_{r,v}$ are the probabilities of a deceleration realization and an acceleration realization, respectively, obtained from the sample data for road type r and speed range v. The other terms in expression (2.6) are as previously defined.

In the remainder of this chapter, we refer to the truncated normalized distribution as the "half-normal distribution."

2.4 Analysis of Results

2.4.1 Comparison of Observed Distributions to Fitted Half-Normal Distributions

The error terms ε obtained upon fitting half-normal distributions (with zero mean and standard deviations obtained from the sample) to the observed values of acceleration and deceleration are shown in Table 2.2 (part a) for the four road types. These error values are satisfactorily low. They support the validity of the hypothesis that accelerations and decelerations are probabilistically distributed, with the fitted distribution in this case being half-normal with zero mean and a standard deviation that decreases as the speed increases. Note that the error term values of deceleration distributions are in most cases lower than those of acceleration distributions because of the availability of more deceleration data and the absence of any acceleration value in the interval $(0,0.2]$. However, error term values are

similar among different road types although there is an uneven distribution of data points among the road types.

Table 2.2*. Error terms of half-normal distributions fitted to the sample acceleration and deceleration distributions, obtained from calibration on all road types (part a) and validation on arterials (part b).

(a)

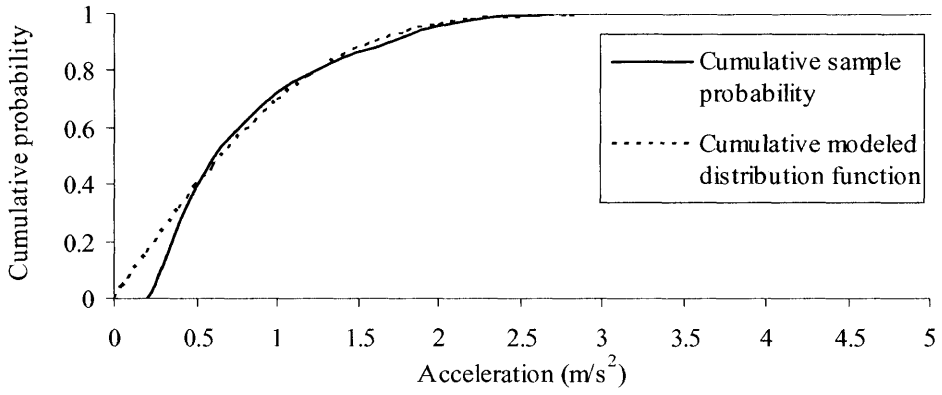
Speed Range (km/h)	Acceleration				Deceleration			
	Interstate Highway	State Highway	Arterial	Collector	Interstate Highway	State Highway	Arterial	Collector
0-10	0.00895	0.00891	0.01106	0.01063	0.00146	0.00120	0.00061	0.00087
11-20	0.00588	0.00844	0.00707	0.00549	0.00560	0.00169	0.00403	0.00218
21-30	0.00374	0.00419	0.00413	0.00601	0.00227	0.00097	0.00102	0.00115
31-40	0.01215	0.01068	0.01507	0.00834	0.01325	0.00934	0.00448	0.03372
41-50	0.01061	0.01659	0.00853	0.02681	0.01494	0.04996	0.03702	0.06297
51-60	0.02055	0.02580	0.02469	0.03926	0.02932	0.04295	0.02982	0.03373
61-70	0.03146	0.03268	0.04044	0.04829	0.02863	0.01082	0.01034	0.01195
71-80	0.04769	0.05109	0.04163	0.05967	0.01719	0.01333	0.00390	0.00350
81-90	0.06232	0.07095	0.07390	0.08689	0.00164	0.00036	0.00129	0.00135
91-100	0.11171	0.10081	0.09852	0.09267	0.00164	0.00132	0.00134	N/A
101-110	0.13349	N/A	N/A	N/A	0.00050	N/A	N/A	N/A
111-120	0.15056	N/A	N/A	N/A	0.00047	N/A	N/A	N/A

* N/A indicates that not enough data were available to calibrate a model for the corresponding speed range and road type.

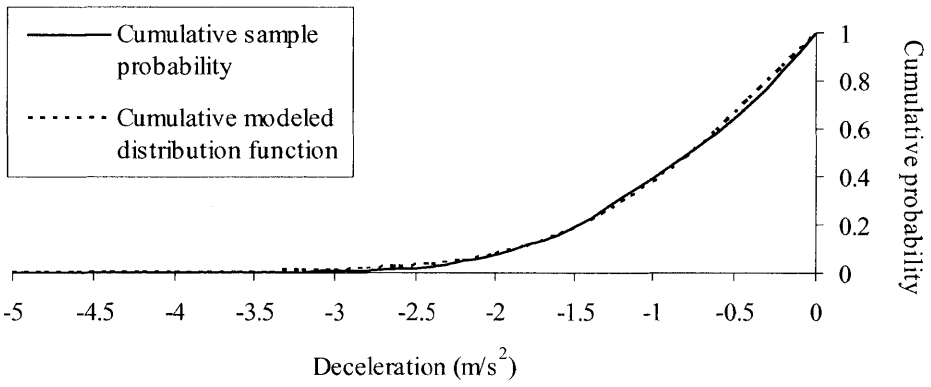
(b)

Speed Range (km/h)	Acceleration	Deceleration
0-10	0.00903	0.00194
11-20	0.01018	0.00265
21-30	0.03033	0.00927
31-40	0.04087	0.04906
41-50	0.05469	0.09007
51-60	0.05512	0.03233
61-70	0.05648	0.00599
71-80	0.07844	0.03284
81-90	0.10668	0.06501

Figure 2.2 shows the cumulative sample probability $F_1(a)$ and the cumulative modeled distribution function $F_2(a)$ for accelerations (part a) and decelerations (part b), respectively, on arterials in subset S_C for the speed range 0-10 km/h. The goodness-of-fit is better for the deceleration data than for the acceleration data due to lack of a sufficient number of observations in the interval $(0,0.2]$, as stated above.



(a)

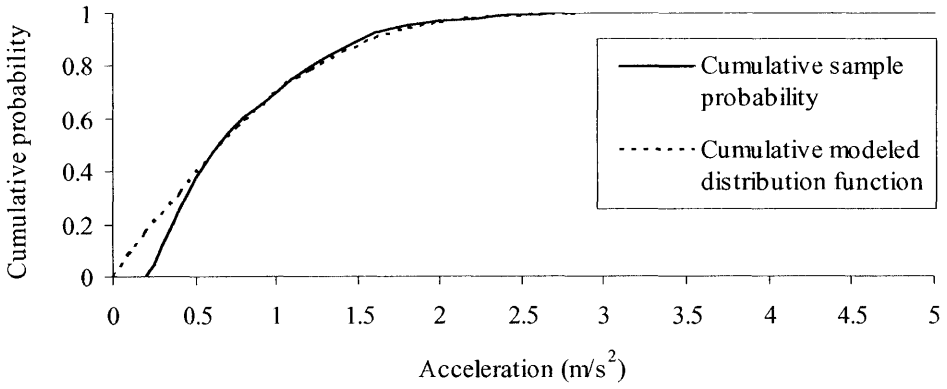


(b)

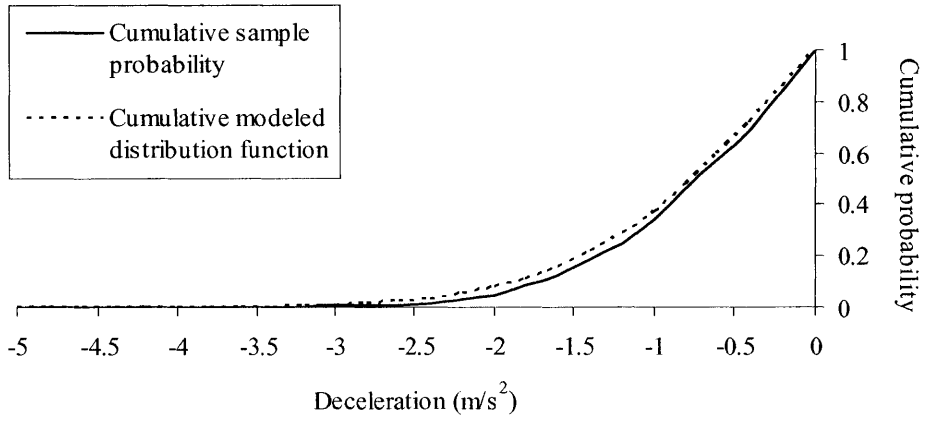
Figure 2.2. Cumulative sample and half-normal distribution functions for the acceleration data (part a) and the deceleration data (part b) used for **calibration** on arterials for the speed range 0-10 km/h.

2.4.2 Validation

Validation of the half-normal distributions was done only for the data of arterials because it contained enough observations to allow for both calibration and validation. For every speed range, the same half-normal distribution that was fitted to the acceleration data in subset S_C was compared to the acceleration data of subset S_V to test whether the distributions developed from a certain sample can be applied to another sample for the same road type. Validation was also done for the deceleration data on arterials. In both cases, the error terms obtained were acceptable, supporting the adoption of probabilistic models to estimate accelerations and decelerations. Figure 2.3 shows the cumulative sample probability of the acceleration data (part a) and deceleration data (part b) in subset S_V and the cumulative distribution corresponding to the modeled density function (which was derived from the calibration data set S_C) on arterials for the speed range 0-10 km/h. The error terms for all speed ranges are shown in Table 2.2 (part b).



(a)



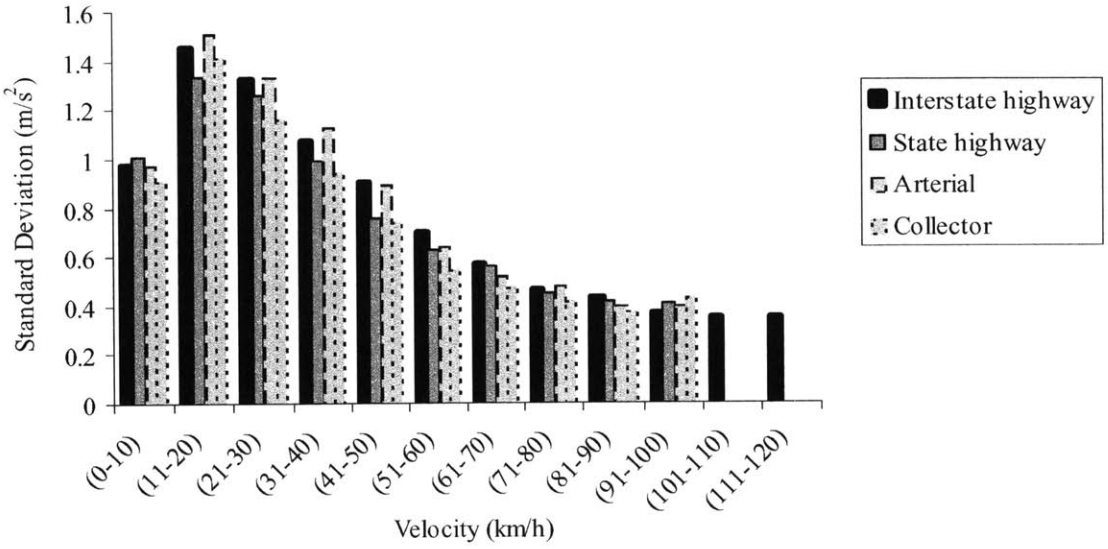
(b)

Figure 2.3. Cumulative sample and half-normal distribution functions for the acceleration data (part a) and the deceleration data (part b) used for **validation** on arterials for the speed range 0-10 km/h.

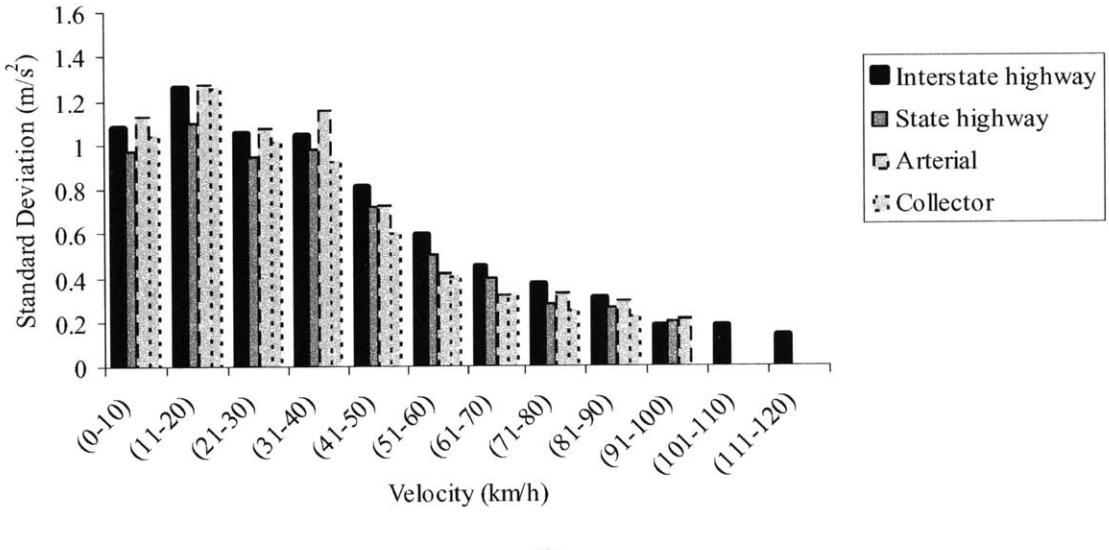
2.4.3 Variation of Standard Deviation of the Distributions with Speed Range and Road Type

Figure 2.4 shows the variation of standard deviation of acceleration (part a) and deceleration (part b) distributions among speed ranges for the four road types. Beyond a certain speed threshold, the standard deviations decrease with speed because at lower speeds, there is a higher probability of achieving high values of acceleration and deceleration than at larger speeds. This diversity of acceleration and deceleration at lower speeds is an important phenomenon in estimating emissions related to engine load or power enrichment. This phenomenon is violated for the first speed range, where the standard deviation increases when moving from 0-10 km/h to 11-20 km/h. The reason for this phenomenon could be that the maximum power available to vehicles at very low speeds is lower than what drivers desire as acceleration. This observation might also be due to the common "lurch" (sudden positive variation in acceleration) that follows a light change, or stop and go traffic, and might be made pronounced by the inexperience of the drivers with a new vehicle throttle. This effect was also observed in LeBlanc et al. (1995).

Road type is seen to have little effect on the variation of acceleration and deceleration distributions, as shown in Figure 2.4. While it is expected that stop and go conditions, characteristic of collectors and arterials, might lead to higher acceleration and deceleration values, the results actually indicate that highways have similar standard deviations to those of arterials and collectors, and in some cases have even higher variations. Note that higher speeds can be reached on highways than on arterials and collectors.



(a)



(b)

Figure 2.4. Variation of standard deviation of acceleration distributions (part a) and deceleration distributions (part b) among different speed ranges and road types.

2.5 Model Application

In this section, we describe a general procedure for a possible application of the acceleration model. Then we depict one instance where the procedure has been applied. This application

has been motivated by the integration of a non-microscopic dynamic traffic model and an instantaneous emission model.

2.5.1 General Procedure

The probabilistic nature of the acceleration model leads to a novel approach for emission modeling. Since accelerations are modeled as random variables, emission factors which are functions of speed and acceleration will themselves be random variables. Therefore, an instantaneous emission model combined with a probabilistic acceleration model can generate, for every road type and speed range, a probabilistic emission distribution from which one can obtain multiple moments of emissions (expected value, standard deviation, etc.).

The approach summarized in the previous paragraph is documented in more details in Cappiello (2002). For a given emission species i, vehicle category c, speed range v, and road type r, an expected emission factor $\bar{e}_{i,c,r,v}$ is calculated based on the probability of occurrence of every acceleration and deceleration, and is given by:

$$\bar{e}_{i,c,r,v} = \underset{a\in[a_1,a_2]}{E}\left[e_{i,c}(v,a)\right] = \int_{a_1}^{a_2} e_{i,c}(v,a)\cdot f^*_{r,v}(a)\,da\,. \tag{2.7}$$

In expression (2.7), $e_{i,c}(v,a)$ is the emission factor, obtained from any instantaneous emission model, for emission species i, vehicle category c, speed v, and acceleration a. a_1 and a_2 are the highest deceleration and acceleration realizations, respectively, in speed range v, as obtained from the sample data. $f^*_{r,v}(a)$ is given by expression (2.6).

The expected emission factor is obtained by discretizing acceleration and deceleration values in the interval $[a_1, a_2]$. Its expression is:

$$\bar{e}_{i,c,r,v} = \sum_{a\in S_a} e_{i,c}(v,a)\cdot \pi_{r,v}(a)\,. \tag{2.8}$$

In the latter expression, $S_a = \{a_1 + h/2, a_1 + 3h/2, \cdots, a_2 - 3h/2, a_2 - h/2\}$ is the discretization interval, and h can be set to any desired value. Here it is set to 0.1 m/s^2. $\pi_{r,v}(a) = \int_{a-h/2}^{a+h/2} f^*_{r,v}(x)\,dx$, which is the probability that the acceleration belongs to the interval

$(a-h/2, a+h/2)$.

This general procedure can be employed in two types of applications. First, it is useful for the integration of non-microscopic traffic models and emission models. In this case, the expected emission factors can be applied to the average speeds which are output by the traffic model in order to predict emissions. A specific application of this type is shown below. Second, the procedure can be used to enhance the accuracy of emission models' predictions in cases where speed is obtained from field measurements, for example through loop detectors, and used as input to the acceleration model which would generate acceleration distributions for a given road type. Any instantaneous emission model would then be able to predict emission distributions (or moments of emissions), given the speed and acceleration (as well as other vehicle and roadway-related factors). Therefore, the acceleration model allows the deployment of more refined and detailed emission models in practice, as it allows the determination of acceleration, which is a quantity not measured in practice, via the measurement of speed only.

2.5.2 Application Example

Below we describe a specific application where expected emission factors have been generated based on speed data obtained from field measurements. The acceleration model has been used in Cappiello (2002) in conjunction with EMIT (EMIssions from Traffic), a recently developed emission and fuel consumption model. We provide a brief description of EMIT, show results of expected emission factors derived from EMIT, and describe the integration of EMIT with a non-microscopic dynamic traffic model.

EMIT is a simple statistical model for instantaneous tailpipe emissions (CO_2, CO, HC, and NO_x) and fuel consumption of light-duty composite vehicles. In order to realistically reproduce the behavior of the emissions, the explanatory variables in EMIT have been derived from the load-based approach, using some simplifying assumptions. The model is calibrated for a set of vehicles driven on standard as well as aggressive driving cycles, and is validated on another driving cycle in order to assess its estimation capabilities. The preliminary results indicate that the model gives reasonable results compared to actual measurements as well as to results obtained with CMEM, a well-known load-based emission

model (see Fig. 2.5). The goodness-of-fit of EMIT varies with different emission species (see Table 2.3), but the model has in general a reasonable predictive accuracy. Furthermore, the model, due to its simple structure, is relatively easy to calibrate and requires less computational time than detailed load-based models. A detailed description of EMIT can be found in Cappiello (2002) and Cappiello et al. (2002).

Table 2.3. R-square (R^2) between the measured and the predicted emission (or fuel consumption) rates from EMIT. Part a: results for calibration. Part b: results for validation. (from (Capiello (2002))).

(a)

	CO_2	CO	HC	NO_X	FR
Engine-out module, category 7	0.98	0.87	0.58	0.86	0.97
Tailpipe module, category 7	0.98	0.84	0.53	0.79	
Engine-out module, category 9	0.97	0.90	0.63	0.87	0.97
Tailpipe module, category 9	0.97	0.88	0.58	0.67	

(b)

	CO_2	CO	HC	NO_X	FR
Engine-out module, category 7	0.96	0.46	0.25	0.83	0.94
Tailpipe module, category 7	0.96	0.36	0.22	0.63	
Engine-out module, category 9	0.95	0.50	0.22	0.83	0.95
Tailpipe module, category 9	0.95	0.43	0.32	0.53	

Expected emission factors have been calculated in advance (offline), according to the general procedure outlined above. The speed data used to compute expected emission factors are obtained from the data set described in this chapter. Figure 2.6 and Figure 2.7 show the calculated expected emission factors of CO_2, CO, HC, and NO_x as well as fuel rates for vehicle category 9 (defined in Cappiello (2002)) as a function of speed on arterials and highways, respectively. In general, the expected emission factor (g/s) increases with speed because of the increase in fuel consumption rate. The expected emission factors are also compared with the facility-specific emission rates from MOBILE6. Note also that expected emission factors would in general be different for different vehicle categories.

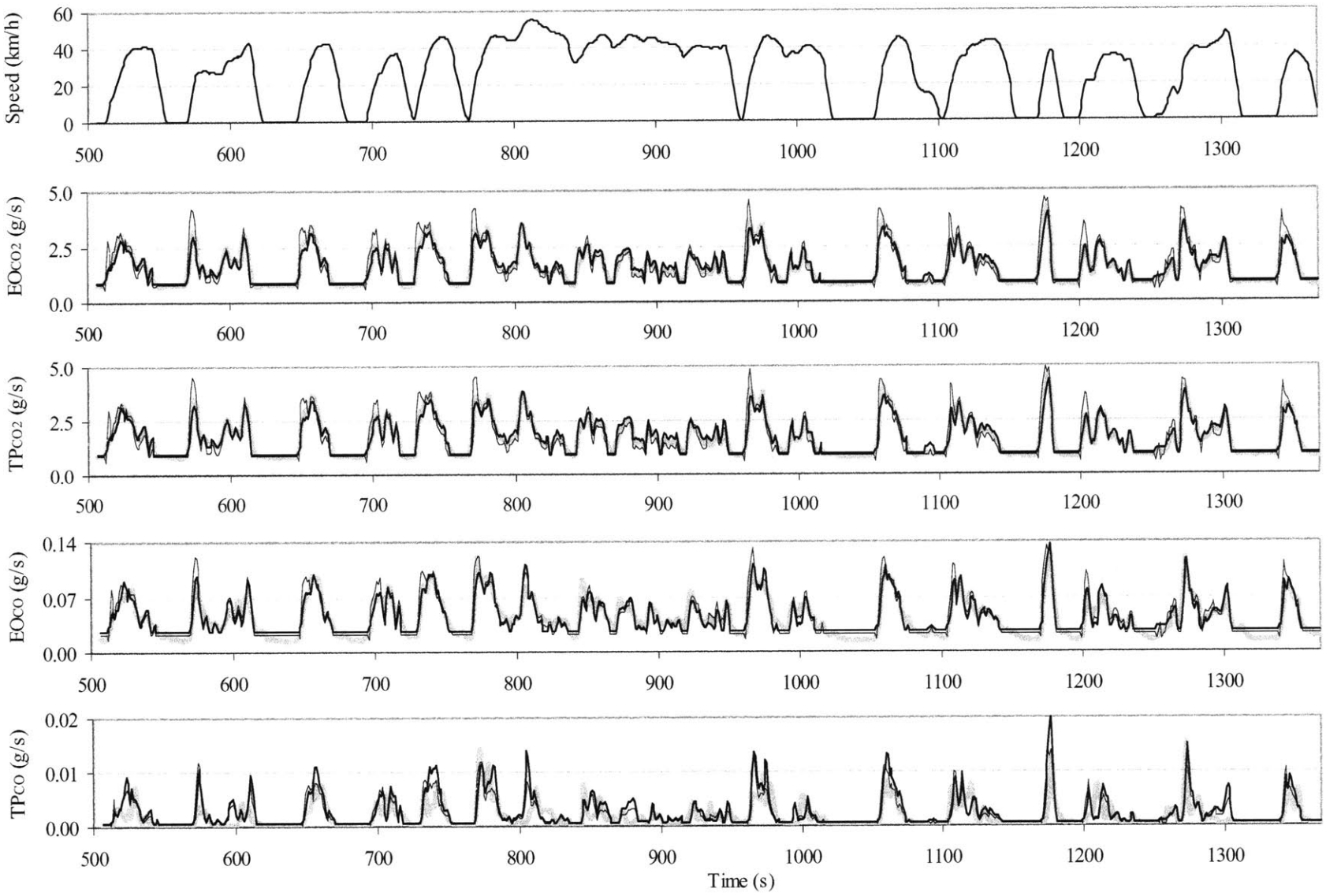


Figure 2.5. Category 9 - FTP bag 2. Second-by-second engine-out (EO) and tailpipe (TP) emission rates of CO_2 and CO. Thick light line: measurements (calibration data); dark line: EMIT predictions; thin line: CMEM predictions. The top plot represents the speed trace. (from (Cappiello (2002))).

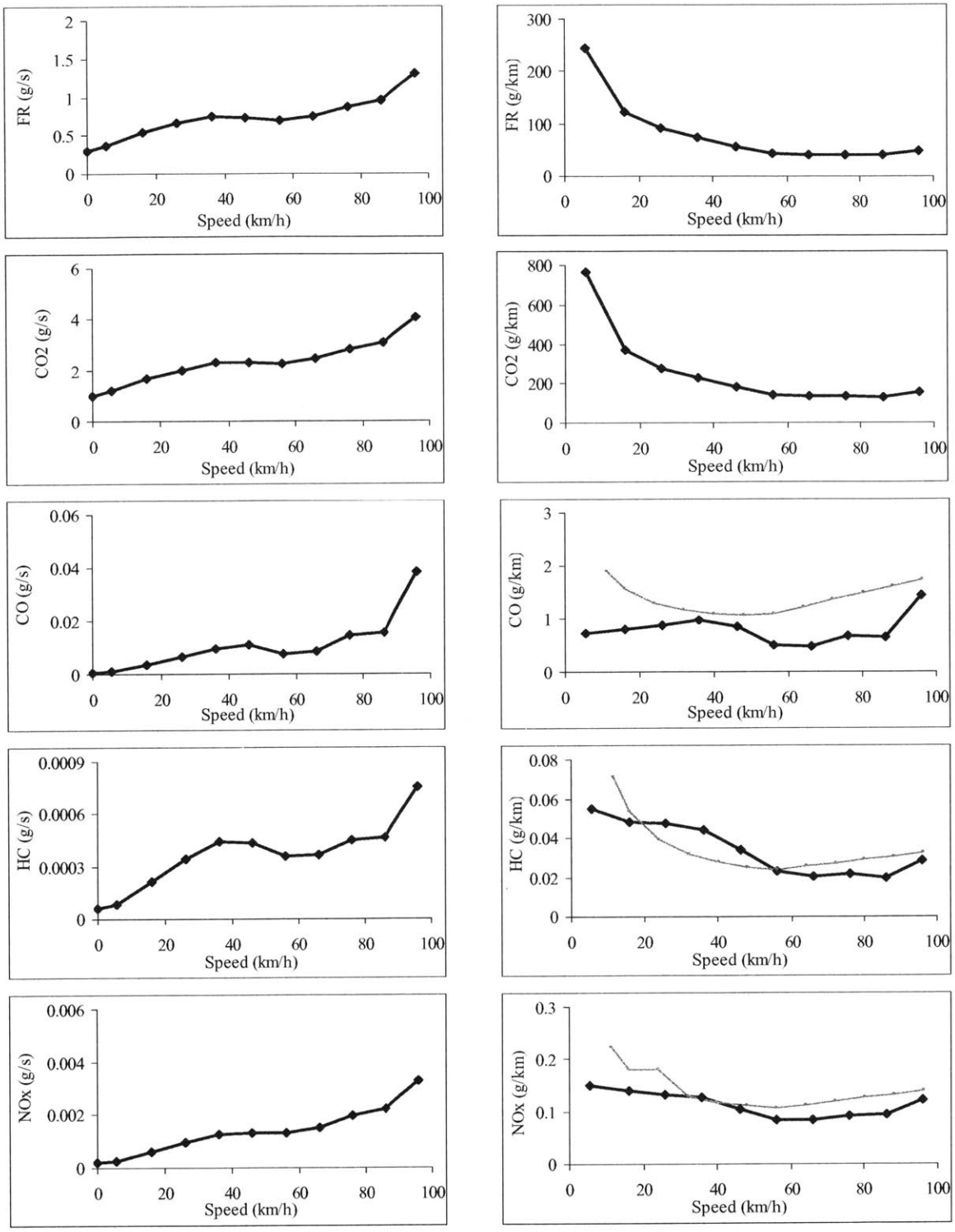


Figure 2.6. Expected emission rates in g/s (on the left) and in g/km (on the right) for road type arterial and vehicle category 9. The expected emission rates in g/km of CO, HC, and NO_X are compared with the facility-specific emission rates from MOBILE6 (thin line). (from (Cappiello (2002))).

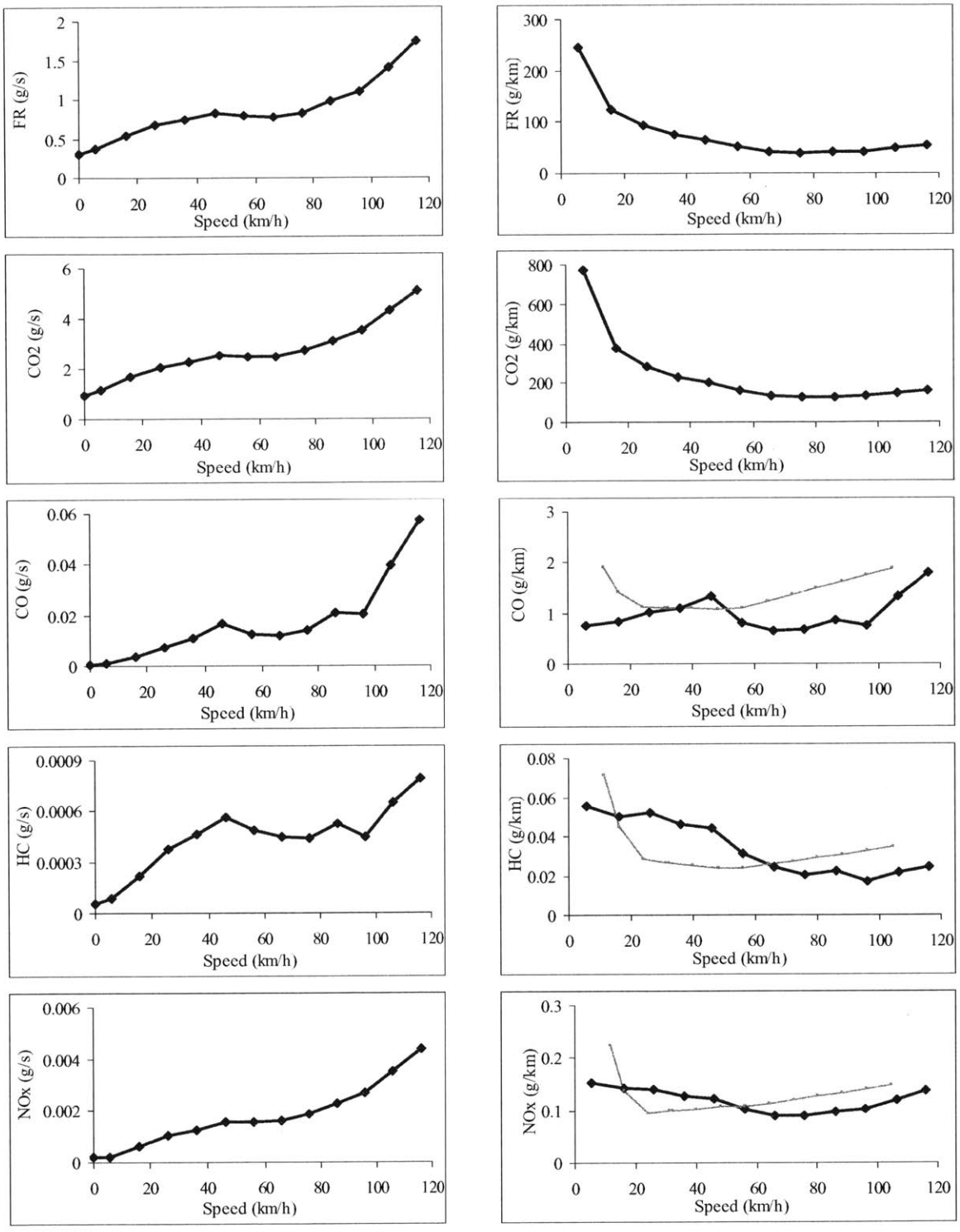


Figure 2.7. Expected emission rates in g/s (on the left) and in g/km (on the right) for road type highway and vehicle category 9. The expected emission rates in g/km of CO, HC, and NO_X are compared with the facility-specific emission rates from MOBILE6 (thin line). (from (Cappiello (2002))).

An integration component is designed to apply the expected emission factors to the output (i.e. time-dependent link speeds) of a mesoscopic traffic model, developed in Bottom (2000), to predict total emissions as well as their spatial and temporal variations. The combined model allows the evaluation of various traffic management strategies and their effectiveness in reducing traffic congestion, air pollution, and fuel consumption. For instance, in Cappiello (2002) various scenarios of traffic conditions (with and without an incident) are considered to assess the impact of dynamic route guidance (an Intelligent Transportation Systems traffic management method) on travel time, emissions, and fuel consumption.

2.6 Conclusions

In this chapter, a probabilistic approach that models acceleration activity as a random variable has been developed. Statistical acceleration and deceleration distributions have been developed as a function of real-world data of vehicle speeds and road types. As the speed range increases, the standard deviation of the acceleration and deceleration distributions decreases because at higher speeds only a limited range of accelerations and decelerations can be achieved due to power and traction limitations. This observation was consistent among all road types. Moreover, the standard deviations are similar among road types, which might suggest that road type has little effect, if any, on acceleration and deceleration variation. However, this effect should be studied further with more data from other cities, since there is reason to believe that driving behavior differs from city to city, especially those that have more hills (LeBlanc et al. (1995)).

For every road type and speed range, and for both acceleration and deceleration, a half-normal distribution having the same mean and standard deviation as the original data was fitted to the observations. The fitted distribution was truncated at some maximum acceleration value in order to consider only physically feasible accelerations. In almost all cases, the fit was very close as indicated by low error term values. This implies that the half-normal distribution well approximates the acceleration and deceleration activity distributions for the given data. The specific parameters of the distribution might have to be calibrated separately for each city since there might be other factors, not captured by

our model, that affect these distributions. Moreover, the distribution that would fit other acceleration and deceleration data from different regions might not be half-normal. However, the same methodology developed in this research can be used to develop other acceleration probability distributions.

A general procedure was given to illustrate an application of the probabilistic modeling approach. Then specific results were provided where the acceleration model was used in conjunction with EMIT (Cappiello (2002), Cappiello et al. (2002)), an instantaneous emission model, to generate expected emission factors for the purpose of integration with a non-microscopic traffic model.

For further research, it would be useful to apply the methodology developed in this research to other data sets (namely the Sierra chase car data) to investigate further the nature of the fitted distributions as well as the effect of road type on these distributions. It would also be important to quantify the activity from freeway ramps. Moreover, it would be interesting to disaggregate this model to assess the impact of driver aggressiveness and vehicle type on the variation of acceleration and deceleration distributions.

Chapter 3

The Minimum Cost Flow Problem in Capacitated Dynamic Networks

3.1 Introduction

The minimum cost flow problem is the problem of sending flows in a network from supply nodes to demand nodes in minimum total cost such that link capacities are not exceeded. This problem has been studied extensively in the context of static networks (Ahuja et al. (1993)). In this chapter, we study the minimum cost flow problem in dynamic (or time-dependent) networks, where link travel times, costs, and capacities are time-varying quantities that depend on the entry time of the link.

The minimum cost flow problem has numerous applications in transportation, logistics, and telecommunication. For instance, in transportation the problem arises in air traffic flow management with enroute (airspace) capacities (Bertsimas and Stock Patterson (1998)) or in road traffic networks with physical or environmental capacities. An application that motivated the work of this chapter has been the optimization of vehicular emissions to meet air quality standards set by regulatory agencies such as the US Environmental Protection Agency (EPA's website: www.epa.gov). These standards mandate that ambient pollutant concentrations in any area of the United States do not exceed a certain threshold beyond which the public and the environment are endangered.

Pollutant concentrations over a link are directly related to the emission rate (emissions mass per unit time) on that link. The emission rate is in turn dependent on the amount of flow on the link. Therefore, by imposing upper bounds on the link flows, one can ensure that no excessive pollutant concentrations (hot spots) are prevalent over any link in the network.

Link cost can be defined in several ways, such as travel time, travel distance, out-of-pocket cost, or emissions generated or inhaled along the link. If link cost is equal to its travel time, for instance, one can exploit some properties of the link travel times as cost functions to develop specialized efficient algorithms for that purpose, as shown later in the chapter. In this chapter, we study the general dynamic minimum cost flow problem, where cost can be equal to any defined criterion. We address one variant of the problem where a given amount of flow needs to be sent from an origin node at a certain departure time, say zero, to a destination node. This is referred to as the one-to-one dynamic minimum cost flow problem. This problem has been studied in Cai et al. (2001), where solution algorithms are developed that consider three particular waiting policies at nodes: no waiting, unbounded waiting, and bounded waiting. Moreover, in Miller-Hooks and Stock Patterson (2002) a solution algorithm that solves the time-dependent minimum time flow problem with unlimited waiting allowed at nodes is given. The main objective of restudying the dynamic minimum cost flow problem in this chapter is to develop more efficient solution algorithms that are obtained by exploiting some properties of the problem.

The approach that we use to solve the minimum cost flow problem is the computation of successive shortest paths in residual networks. We note that in the shortest path problem, which is a particular case of the minimum cost flow problem, link capacities can be viewed as infinite; thus, all the flow can be sent on a shortest path. However, in the general capacitated minimum cost flow problem, a series of shortest path problems should be solved since a single shortest path might not have enough capacity to carry all the flow. For each of the successive shortest paths, an amount of flow equal to the path capacity is augmented until all the flow has been sent from the origin to the destination. Moreover, while dynamic shortest path problems can be viewed as static shortest path problems in a static acyclic time-space network (defined in Section 3.2.2), the residual

time-space network (defined in Section 3.2.3) in which the minimum cost flow problem is solved is not acyclic along the time dimension. Thus, to solve the minimum cost flow problem, there is need to develop solution algorithms that are not specific for acyclic time-space networks, as opposed to classical dynamic shortest path algorithms in dynamic networks. The reader interested in the latter topic is referred to Grier and Chabini (2002), Chabini and Lan (2002), Pallottino and Scutellà (1998), Chabini (1998), Cai et al. (1997), Ziliaskopoulos (1994), Kaufman and Smith (1993), Orda and Rom (1990), Dreyfus (1969), and Cooke and Halsey (1966).

We present two algorithms, denoted as Algorithm B and Algorithm C, for the shortest path computation involved in the solution of the dynamic minimum cost flow problem. We also review an algorithm, due to Cai et al. (2001), which we denote as Algorithm A. Algorithms A, B, and C were implemented, and their computational efficiencies were assessed by using large-size capacitated dynamic networks. The computational results indicate that Algorithms B and C are more efficient than Algorithm A. Moreover, for the test networks used, the successive shortest path algorithm employing Algorithm C achieved significant time savings, compared to that employing Algorithm A (by up to a factor of 113) and that employing Algorithm B (by up to factors of 25, 39, and 72 for three different implementations of Algorithm B).

The remainder of this chapter is organized as follows. In Section 3.2, we provide definitions and notation. In Section 3.3, we review a formulation of the dynamic minimum cost flow problem. In Section 3.4, we describe a generic solution algorithm and present some properties that will be useful for developing solution algorithms. We review an algorithm developed in Cai et al. (2001) for the computation of minimum cost paths in the residual network, and develop two efficient minimum cost path algorithms. In Section 3.5, we discuss the case where link travel cost is equal to its travel time and describe specialized algorithms that compute paths which minimize total travel times in capacitated dynamic networks. Then we describe how the developed algorithms can be used or extended to account for various waiting policies, multiple sources, destinations, and departure times. Finally in Section 3.6, we provide experimental results for the solution algorithms presented in this chapter.

3.2 Notation and Definitions

3.2.1 Network Data

Let $G=(N,A)$ be a directed network. $N=\{1,...,n\}$ is the set of nodes and $A=\{1,...,m\}$ is the set of arcs. Let $A(i)$ represent the set of nodes after node i, i.e. $A(i)=\{j:(i,j)\in A\}$. We refer to $A(i)$ as the forward star of i. Let $B(j)$ represent the set of nodes before node j, i.e. $B(j)=\{i:(i,j)\in A\}$. We refer to $B(j)$ as the backward star of j. We associate with every arc (i,j) a positive travel time $d_{ij}(t)$, a travel cost $c_{ij}(t)$, and a non-negative capacity $U_{ij}(t)$, where $t\in T=\{0,1,2,...,M-1\}$ is the entry time of the link, and M is a time horizon beyond which travel is prohibited. The capacity $U_{ij}(t)$ refers here to the maximum flow that can be admitted at the entrance of arc (i,j) at time t. In certain applications though, the capacity of an arc might refer to its storage capacity, i.e. the maximum number of flow units that can be present on the arc at any time. Every arc (i,j) has a residual capacity $r_{ij}(t)$ defined as: $r_{ij}(t)=U_{ij}(t)-f_{ij}(t)$, where $f_{ij}(t)$ is the flow measured at the entrance of arc (i,j) at time t. Denote the source node by s and the destination node by q. Let v be the amount of flow that should be sent from s to q. We assume that all network data is deterministic and that no waiting is allowed at nodes in the general model. Later in the chapter, we show how the solution algorithms that we develop for the no-waiting case can be extended to allow for general waiting policies at the nodes.

3.2.2 Time-Space Network

The time-space network is a useful tool for implicitly studying the minimum cost flow problem. Note that although we exploit the concept of time-space network to illustrate how the different solution algorithms operate, none of the algorithms presented in this chapter was implemented by explicitly constructing the time-space network.

The time-space network is a static network constructed by expanding the original network in the time dimension by making a separate copy of every node $i\in N$ at every time $t\in T$, called a node-time pair (i,t). Let $G^*=(N^*,A^*)$ represent the time-expanded

network of the original dynamic network. Set N^* denotes the nodes of G^*, which are given by $N^* = \{(i,t): (i,t) \in N \times T\}$. The set of arcs A^* consists of arcs from every node-time pair $(i,t) \in N^*$ to every other node-time pair $(j, t + d_{ij}(t))$, where $j \in A(i)$ and $t + d_{ij}(t) \leq M$. The cost of an arc connecting (i,t) to $(j, t + d_{ij}(t))$ is equal to $c_{ij}(t)$. We note that the time-space network is acyclic along the time dimension.

3.2.3 Dynamic Time-Dependent Residual Network

We rely on the concept of residual networks to develop solution algorithms for the minimum cost flow problem. A definition of residual networks is provided in Ahuja et al. (1993) for static networks and in Cai et al. (2001) for dynamic networks. The dynamic residual network corresponding to a feasible flow f can be viewed as the static residual network of the time-space network corresponding to the dynamic network, and is derived as follows. We denote the reverse arc of an arc (i, j) as $\langle j, i \rangle$. Every arc in the time-space network, connecting node-time pair (i,u) to node-time pair (j,t), and on which $f_{ij}(u)$ flow units depart its beginning at time u, has a corresponding reverse arc connecting (j,t) to (i,u) such that: $d\langle j,i,t,u \rangle = -d_{ij}(u)$, $c\langle j,i,t,u \rangle = -c_{ij}(u)$, and $r\langle j,i,t,u \rangle = -r_{ij}(u)$, where $d\langle j,i,t,u \rangle$, $c\langle j,i,t,u \rangle$, and $r\langle j,i,t,u \rangle$ are the travel time, travel cost, and residual capacity of the reverse arc connecting (j,t) to (i,u), if one "departs" along this reverse arc at time t and arrives at its end at time u. Sending flow on a reverse arc is equivalent to reducing flow on its corresponding forward arc which carried flow in a previous iteration.

We define an augmenting path P from source node s to destination node q in the residual network as a path which has positive residual capacity $U(P)$, which is equal to the minimum residual capacity of its constituent arcs. Assume that an augmenting path P connects the nodes $s = i_0, i_1, ..., i_q = q$. Let t_i denote the arrival time at a node i, which is also equal to the departure time from i if waiting is not allowed at nodes. We define the following recursive relationship: $t_{i_0} = 0$, $t_{i_k} = t_{i_{k-1}} + d_{i_{k-1}i_k}(t_{i_{k-1}})$ if (i_{k-1}, i_k) at time $t_{i_{k-1}}$ is

not a reverse arc, and $t_{i_k} = t_{i_{k-1}} + d\langle i_{k-1}, i_k, t_{i_{k-1}}, t_{i_k} \rangle$ otherwise. We denote the predecessor node-time pair $(i_{k-1}, t_{i_{k-1}})$ of node-time pair (i_k, t_{i_k}) along path P as $p(i_k, t_{i_k})$.

3.3 Formulation

In this section, we review a formulation of the minimum cost flow problem in discrete dynamic networks. This formulation is given in Cai et al. (2001).

Min $\sum_{(i,j)\in A} \sum_{t\in T} c_{ij}(t) f_{ij}(t)$

Subject to:

$$\sum_{(s,i)\in A} f_{si}(0) = v \tag{3.1}$$

$$\sum_{(i,j)\in A, t'+d_{ij}(t')=t} f_{ij}(t') - \sum_{(j,k)\in A} f_{jk}(t) = 0 \quad \forall j \in N \setminus \{s,q\}, t \in T \tag{3.2}$$

$$\sum_{(i,q)\in A} \sum_{\{t \mid 0\le t\le M, t+d_{iq}(t)\le M\}} f_{iq}(t) = v \tag{3.3}$$

$$0 \le f_{ij}(t) \le U_{ij}(t) \quad \forall (i,j) \in A, t \in T \tag{3.4}$$

The objective function is to minimize the total travel cost of the flow traveling on all the links of the network before the time horizon M. Constraints (3.1), (3.2), and (3.3) are the flow conservation constraints. Constraint (3.1) ensures that the total flow departing the origin node s at time zero is equal to its supply v. Constraints (3.2) ensure that the total flow arriving at an intermediate node j at time t is equal to the flow that departs j at time t. Constraint (3.3) states that the total flow arriving at the destination node q is equal to its demand v. Finally constraints (3.4) are the flow bound constraints for each link at each time.

3.4 Solution Algorithms

In this section, we present solution algorithms that solve the one-to-one dynamic minimum cost flow problem. Extensions to the problem, including multiple sources, destinations, and departure times and various waiting policies, are addressed in Section 5 of this chapter.

We use the well-known successive shortest path approach, which we adapt to the dynamic residual network, to develop solutions algorithms for the formulation presented above. As noted previously, the time-space network serves as an implicit tool for studying the problem and interpreting the various algorithms, but is not explicitly constructed. The successive shortest path approach is a classical approach that can be found in textbooks on network flow algorithms (see for instance Ahuja et al. (1993)). The approach adapted to the dynamic residual network is based on solving a series of successive shortest path problems, where each is solved in a residual time-space network. An amount of flow equal to the capacity of each minimum cost path obtained is augmented, until all the flow has been sent from the origin to the destination. Algorithms developed for the dynamic minimum cost flow problem are specializations of this approach. The main difference among the algorithms is the algorithm used to solve a shortest path problem in the dynamic residual network. Below we describe the steps used in the successive shortest path algorithm adapted to solve the dynamic minimum cost flow problem. Then we define some properties that will be useful to develop various solution algorithms with various levels of efficiency.

Let z be the amount of flow which has been sent so far by the algorithm. Initially z is set to zero, as no flow has been augmented yet. When z is equal to the given amount of flow v that should be sent from s to q, the algorithm is terminated. Let $\pi(q)$ represent the cost of a minimum cost path from s to q obtained in a certain augmentation iteration, and let $\tau(q)$ represent the travel time of this path. If $\tau(q)$ is greater than the time horizon M, the problem is infeasible and the algorithm is terminated. Let $U'(P)$ denote the minimum of two terms: the residual capacity of path P and the amount of flow that still needs to be sent from s to q. The generic structure of the successive shortest path algorithm which solves the problem is described as follows:

Step 1: Initialization

$z = 0$

Step 2: Compute a minimum cost path with positive residual capacity from the origin to the destination

(Algorithms to perform this step are given later in this section).

Step 3: Find the capacity $U(P)$ of the augmenting path

If $\tau(q) > M$, return, problem is not feasible, and stop.

Otherwise, $U(P) = \min\left\{\min_{(i,j)\in P} r_{ij}(t_i), \min_{\langle i,j\rangle\in P} r\langle i,j,t_i,t_j\rangle\right\}$, and $U'(P) = \min\{v - z, U(P)\}$.

Step 4: Augment flow and update the dynamic residual network

For every arc on the augmenting path connecting node-time pair (i,t_i) to node-time pair (j,t_j)

If $t_j > t_i$, then $r_{ij}(t_i) = r_{ij}(t_i) - U'(P)$ and $r\langle j,i,t_j,t_i\rangle = r\langle j,i,t_j,t_i\rangle + U'(P)$

Otherwise, $r_{ji}(t_j) = r_{ji}(t_j) + U'(P)$ and $r\langle i,j,t_i,t_j\rangle = r\langle i,j,t_i,t_j\rangle - U'(P)$

$z = z + U'(P)$

If $z = v$, then stop. Otherwise, go to Step 2.

Finding the capacity of a minimum cost path, augmenting the flow, and updating the residual network (Steps 3 and 4) are standard procedures common to all solution algorithms presented in this chapter and the solution algorithms known in the literature. The method of computing minimum cost paths is however peculiar to every solution algorithm. We present three such algorithms for minimum cost path computations. As mentioned previously, the first algorithm for minimum cost path computations, denoted as Algorithm A, is an existing algorithm, developed in Cai et al. (2001). Algorithm A is interpreted in this chapter differently than in Cai et al. (2001). The second and third algorithms, denoted respectively as Algorithm B and Algorithm C, are more efficient solution algorithms developed by progressively exploiting some properties of the time-space network. Let $\pi(i,t)$ denote the minimum travel cost from the source $(s,0)$ to node-time pair (i,t), and let $\hat{\pi}(i,t)$ denote an upper bound on this cost. That is, $\hat{\pi}(i,t)$ is the cost of a minimum cost path found so far by the algorithm from node-time pair $(s,0)$ to node-time pair (i,t). Before presenting the three approaches, we make some observations that will be useful in the development of the solution algorithms:

1. We note that the residual time-space network is composed of two subnetworks: a forward network, denoted as G^+, consisting of the set of forward arcs, denoted as A^+, that have positive travel times; and a reverse network, denoted as G^-, consisting of the set of reverse arcs, denoted as A^-, that have negative travel times. Each of the two subnetworks G^+ and G^-, alone, is acyclic. There are two approaches to visit the residual time-space network to compute minimum cost paths. The first approach is to visit the two subnetworks successively, making use of the acyclicity property. In this approach, the forward subnetwork G^+ is visited first, and minimum cost labels at the nodes are computed. Next the reverse subnetwork G^- is visited to update the minimum cost labels at the nodes, using as initial values the labels computed from the subnetwork G^+. The forward and reverse subnetworks are visited successively until the minimum cost labels at all nodes, as obtained from both subnetworks, are equal. At this point a minimum cost path with positive residual capacity has been found. The second approach to compute minimum cost paths is to visit the two subnetworks G^+ and G^- simultaneously rather than using the results of one subnetwork to initialize the labels in the other subnetwork. We present one algorithm that follows the first approach and two algorithms that follow the second approach later in this section.
2. In general, a node-time pair (i,t) might be visited more than once by a solution algorithm because different paths reaching (i,t) will in general produce different cost labels at (i,t). For example, suppose that the cost label of (i,t) obtained from a path P_1 that reaches i at time t is equal to $\hat{\pi}_{P_1(i,t)}$. And suppose that after (i,t) is visited, its cost label decreases to $\hat{\pi}_{P_2(i,t)}$ due to another path P_2 that reaches i at time t. Then (i,t) should be revisited by the algorithm as its decreased cost label could potentially update the cost labels of nodes in its forward star. Consequently, an arc connecting node-time pair (i,t_i) to node-time pair (j,t_j) might be visited more than once in every minimum cost path computation. Note the difference here between the general minimum cost flow problem and one of its variants, the minimum time flow problem, where a link cost is equal to its travel time. In the latter case, a node-time

pair (i,t_i) needs to be visited at most once because if it can be reached, its cost label, which is the travel time from $(s,0)$ to (i,t_i), will always be equal to t_i irrespective of the path through which (i,t_i) is reached. Therefore, every arc connecting node-time pair (i,t_i) to node-time pair (j,t_j) needs to be visited at most once in every minimum time path computation. The latter problem is a connectivity problem between the source node and all nodes that correspond to the destination node in the time-space network.

3. We note that it is not necessary to visit all nodes in the time-space network. Only the relevant node-time pairs that can be reached (or a subset of these, as will be shown later in the chapter) along paths departing node-time pair $(s,0)$ need to be visited.

Below we describe the procedures of exploring the network and finding a minimum cost path in Algorithms A, B, and C (i.e. Step 2 of the generic algorithm).

3.4.1 Algorithm A

We review an existing algorithm, developed in Cai et al. (2001), for minimum cost path computations involved in the solution algorithm to the dynamic minimum cost flow problem. We present below an interpretation of this algorithm that is different from the description given in Cai et al. (2001), which we refer to as Algorithm A, in the time-space network, and according to the properties that were discussed above. Algorithm A employs the first approach outlined above to compute minimum cost paths. That is, the two acyclic subnetworks G^+ and G^- are explored separately. In each subnetwork, the algorithm visits all node-time pairs (j,t) and computes the estimated minimum cost labels $\hat{\pi}(j,t)$ according to the following optimality conditions:

$$\hat{\pi}(j,t)^k = \min\left\{\hat{\pi}(j,t)^{k-1}, \min_{\{i|(i,j)\in A^+\}}\left\{\min_{\{u|u+d_{ij}(u)=t\ \&\ r_{ij}(u)>0\}}\left\{\hat{\pi}(i,u)^k + c_{ij}(u)\right\}\right\}\right\} \tag{3.5}$$

$$\hat{\pi}(j,t)^{k+1} = \min\left\{\hat{\pi}(j,t)^k, \min_{\{i|\langle i,j\rangle\in A^-\}}\left\{\min_{\{u|u+d\langle i,j,u,t\rangle=t\ \&\ r\langle i,j,u,t\rangle>0\}}\left\{\hat{\pi}(i,u)^{k+1} + c\langle i,j,u,t\rangle\right\}\right\}\right\}, \tag{3.6}$$

where k denotes the iteration number, i.e. the number of subnetworks explored so far (k is referred to as a section index in Cai et al. (2001)). Equations (3.5) and (3.6) are used in subnetworks G^+ and G^-, respectively. Exploration of subnetwork G^+ is done in increasing order of time as G^+ is acyclic (Chabini (1998), Pallottino and Scutellà (1998)). The estimated minimum cost label $\hat{\pi}(j,t)^k$ in iteration k is initialized to the value it had in iteration $(k-1)$, and is updated by exploring all the node-time pairs (i,u) in the backward star of (j,t), even though some of these node-time pairs may never be reached from $(s,0)$. When all node-time pairs in subnetwork G^+ have been visited, a new iteration $(k+1)$ begins, and subnetwork G^- is explored in decreasing order of time (corresponding to the reverse arcs), using $\hat{\pi}(j,t)^k$ as initial values of the cost labels of all nodes (j,t). Equations (3.5) and (3.6) are applied successively until all cost labels obtained from one subnetwork are equal to the cost labels obtained from the other subnetwork. When the algorithm terminates, $\pi(i,t)=\hat{\pi}(i,t), \forall (i,t)\in N\times T$. The travel cost of a minimum cost path is given by $\pi(q)=\min_{t\in T}\{\pi(q,t)\}$, and its travel time is given by $\tau(q)=Arg\min_{t\in T}\{\pi(q,t)\}$. The running time complexity of the dynamic minimum cost flow algorithm based on Algorithm A is in $O((nmM^2)v)$, since at most v augmentations are done by the algorithm. In each augmentation, at most nM iterations (sections) are done, and in each iteration mM arcs are explored (which corresponds to the number of arcs in the time-space network).

To argue the correctness of Algorithm A, we note that Algorithm A applies an identical logic as that used in Yen's implementation (1970) of Bellman-Ford's shortest path algorithm, and whose run time is equal to half that of Bellman-Ford. Yen's algorithm divides the network G into two subnetworks, G_1 and G_2. G_1 consists of all arcs directed from a node i to a node j, where $i<j$, and G_2 consists of all remaining arcs, i.e. those connecting a node i to a node j, where $i>j$. Note that both subnetworks G_1 and G_2 are acyclic. The algorithm computes estimates of minimum cost labels in one subnetwork and uses these estimates to initialize the cost labels in the other subnetwork.

This procedure is repeated successively until all the estimated minimum cost labels from both subnetworks converge to the same value. It can thus be seen that Algorithm A that operates on the two acyclic subnetworks G^+ and G^- is identical to the algorithm developed in Yen (1970).

3.4.2 Algorithm B

Algorithm B differs from Algorithm A in several respects. First, the minimum cost labels of the nodes are obtained by simultaneously exploring the forward and reverse arcs, rather than exploring the two subnetworks separately. Second, instead of computing the minimum cost labels of all nodes in the time-space network, only the relevant nodes that could be reached from the origin node at departure time zero are visited. Third, the nodes are visited in increasing order of time (as illustrated next) which leads to a smaller number of arc explorations. Next we describe the algorithm and provide its pseudocode.

3.4.2.1 Description of Algorithm B

We maintain a set of candidate nodes C which initially includes only the source node at departure time zero, i.e. $(s,0)$. The set C holds all node-time pairs (i,t) which have been reached so far by the algorithm, and which are to be visited. Note that a node-time pair (i,t) might be inserted in the set C more than once, as explained previously. We initialize the cost labels $\hat{\pi}(i,t)$ of all node-time pairs (i,t) to infinity, and the minimum cost label $\hat{\pi}(s,0)$ (which is equal to $\pi(s,0)$) of $(s,0)$ to zero.

The algorithm visits nodes in increasing order of time, taking into consideration the existence of reverse arcs with negative travel times. We define a time-bucket B_t as an array or a list that stores the nodes which have been reached at time t, i.e. node-time pairs (i,t). Therefore, we maintain $(M+1)$ buckets each corresponding to one arrival time t at the nodes, $0 \le t \le M$. We always select elements from the minimum time bucket B_t. When the minimum time bucket B_t is empty, we check if any elements have been inserted at a lower time bucket $B_{t'}$, where $t' < t$ (due to the reverse arcs). If so, we next select those elements in the minimum non-empty time bucket $B_{t'}$. Otherwise, we

select elements from the next non-empty time bucket $B_{t''}$, where $t'' > t$. The rationale behind this selection strategy is to achieve computational time savings by visiting those node-time pairs that are reached along reverse arcs as soon as they are reached, as these can potentially update the cost labels of node-time pairs in higher time buckets. Delaying their visit, on the other hand, could result in revisiting several node-time pairs in higher time buckets.

For every node-time pair (i,t) selected from B_t, we explore the arcs with positive residual capacity that connect (i,t) to node-time pairs (j,u), where $0 < u = t + d_{ij}(t) \leq M$ if the arc connecting (i,t) to (j,u) is a forward arc and $0 \leq u = t - d_{ji}(u) < M$ if it is a reverse arc. We update the cost label $\hat{\pi}(j,u)$, if necessary, and add (j,u) to bucket B_u of the candidate set C if it is not already in B_u. We repeat this process until there are no more candidate nodes in C. When the algorithm terminates, $\pi(i,t) = \hat{\pi}(i,t), \forall (i,t) \in N \times T$. The travel cost of a minimum cost path is given by $\pi(q) = \min_{t \in T}\{\pi(q,t)\}$, and its travel time is given by $\tau(q) = Arg\min_{t \in T}\{\pi(q,t)\}$.

Note that other implementations of the candidate set C and other selection functions are also possible. Below we provide the pseudocode of Algorithm B for a general implementation of the candidate set C.

3.4.2.2 Pseudocode of Algorithm B

Step 1: Initialization

$\hat{\pi}(i,t) = \infty, \forall (i,t) \in N \times T; \hat{\pi}(s,0) = 0$

$\pi(q) = \infty; C = \{(s,0)\}$

Step 2: Node selection

Select (i,t_i) from C; $C = C \setminus \{(i,t_i)\}$

Step 3: Explore forward and backward star

For all $j \in A(i)$ such that $r_{ij}(t_i) > 0$ and $j \neq s$

$$t_j = t_i + d_{ij}(t_i)$$

$$\text{If } \left(t_j \leq M \text{ and} \left(\hat{\pi}(i,t_i) + c_{ij}(t_i) < \hat{\pi}(j,t_j) \right) \right) \text{ then}$$

$$\hat{\pi}(j,t_j) = \hat{\pi}(i,t_i) + c_{ij}(t_i)$$
$$p(j,t_j) = (i,t_i)$$
$$\text{If } (j,t_j) \notin C, \text{ then } C = C \cup \{(j,t_j)\}$$

For all $j \in B(i)$ such that $j \neq s$

For all t_j such that $t_i = t_j + d_{ji}(t_j)$ and $r_{ji}(t_j) < U_{ji}(t_j)$

$$\text{If } \left(\hat{\pi}(i,t_i) - c_{ji}(t_j) < \hat{\pi}(j,t_j) \right) \text{ then}$$

$$\hat{\pi}(j,t_j) = \hat{\pi}(i,t_i) - c_{ji}(t_j)$$
$$p(j,t_j) = (i,t_i)$$
$$\text{If } (j,t_j) \notin C, \text{ then } C = C \cup \{(j,t_j)\}$$

Step 4: Stopping criterion

If $C = \phi$, STOP; otherwise, go to Step 2.

Note that in Step 3, if the origin node s is reached, it is not added to the candidate set so as to restrict the departure time at s to $t = 0$.

3.4.2.3 Implementation Details

The selection function in Step 2 of the pseudocode of Algorithm B will lead to multiple implementations of the candidate set C, such as a time-bucket, a queue, or a dequeue. The number of node-time pairs visited by the algorithm (and inserted in set C) will in general be different for different data structures since the order of node additions and selections is not the same for all data structures, and the number of label revisions before convergence is a function of this order. Consequently, the theoretical running time complexity of the algorithm depends on the data structure used to implement the

algorithm. When the candidate set is implemented as a queue or a dequeue, Algorithm B can be viewed as a direct application of a static label-correcting shortest path algorithm to the residual time-space network, and hence upper bounds can easily be established on its running time. Note that the running time complexity of Algorithm B with a time-bucket implementation of the candidate set is still under investigation as this implementation differs from all known label-correcting implementations in the literature. However, as the numerical results in Section 3.6 indicate, the time-bucket implementation is more efficient in practice than both the queue and the dequeue implementations.

3.4.3 Algorithm C

Algorithm C aims at reducing significantly the number of node-time pairs that need to be visited in Algorithm B by employing special node addition and selection procedures. The basic idea is to compute lower bounds on the minimum travel cost from any node-time pair to the destination, and utilize these bounds to achieve computational time savings by reducing the search area in the dynamic residual network. Let $e_n(i,t)$ be the minimum travel cost from node i to destination node q, if one departs node i at time t after the n^{th} augmentation and update of the residual network. We first state a property related to the lower bounds, which will be needed in developing the solution algorithm.

Lemma 3.1: *The function* $e_n(i,t)$ *is a non-decreasing function of the number of augmentations. That is:* $e_{n-1}(i,t) \leq e_n(i,t)$.

The proof of Lemma 3.1 is given in Appendix A. ■

Let $\hat{e}(i,t)$ be a lower bound on the minimum travel cost from node-time pair (i,t) to the destination node q. Since the minimum cost labels $e_n(i,t)$ are non-decreasing functions of the number of augmentations, one can use $e_0(i,t)$ (the minimum cost from (i,t) to q obtained before augmenting any flow) as an estimate of $\hat{e}(i,t)$. The lower bounds $\hat{e}(i,t)$ can be used to improve the node selection and label update procedures as follows.

Let $\lambda(i,t)$ be the minimum travel cost among all paths from origin node s at departure time zero to the destination q constrained to go through node-time pair (i,t). For an estimate $\hat{\lambda}(i,t)$ of $\lambda(i,t)$, one can use the sum of travel cost from $(s,0)$ to (i,t) and the lower bound on the travel cost from (i,t) to q, i.e. $\hat{\lambda}(i,t)=\hat{\pi}(i,t)+\hat{e}(i,t)$. $\hat{\lambda}(i,t)$ could then be updated if the cost label $\hat{\pi}(i,t)$ is updated. We note that if $\hat{\lambda}(i,t)$ is greater than or equal to the minimum cost label $\pi(q)$ at the destination found by the algorithm so far, it is not useful to explore the forward and backward star of node-time pair (i,t), since the minimum cost label at the destination from all these elements is greater than or equal to $\hat{\lambda}(i,t)$. That is, for all (j,u) in the forward star or backward star (along reverse arcs with positive capacity) of (i,t): $\hat{\lambda}(j,u)\geq\hat{\lambda}(i,t)$ and since $\hat{\lambda}(i,t)\geq\pi(q)$, then $\hat{\lambda}(j,u)\geq\pi(q)$. Therefore, none of the nodes in the forward or backward star of (i,t) can improve the minimum cost label at the destination. The use of this observation considerably reduces the number of nodes that are visited by the algorithm, which is described next.

3.4.3.1 Description of Algorithm C

To compute the lower bounds $\hat{e}(i,t)$ on the minimum travel costs for all $(i,t)\in N\times T$, one can use any dynamic all-to-one shortest path algorithm, such as algorithm DOT developed in Chabini (1998). In this case, $\hat{e}(i,t)=e_0(i,t)$, which is the minimum cost obtained before augmenting any flow. Note that the efficiency of Algorithm C depends on the quality of the lower bounds $\hat{e}(i,t)$ chosen. Values of $\hat{e}(i,t)$ smaller than $e_0(i,t)$ can be used. The computation of those lower bounds will take less time than that of $e_0(i,t)$, but they would not reduce the set of node-time pairs searched by the algorithm as would $e_0(i,t)$. For example, static lower bounds can be obtained from a virtual static network where link travel costs are defined as: $c_{ij}=\underset{t\in T}{\text{Min}}\{c_{ij}(t)\}$. In the extreme case, one

can use $\hat{e}(i,t)=0$, $\forall(i,t)\in N\times T$, in which case Algorithm C would correspond to Algorithm B with the candidate set implemented as a cost-bucket (where nodes are selected from the candidate set in increasing order of cost).

After computing the lower bounds, we set the labels $\hat{\lambda}(i,t)$ and $\hat{\pi}(i,t)$ of all node-time pairs (i,t) to infinity, and we set $\hat{\pi}(s,0)$ to zero and $\hat{\lambda}(s,0)$ to $\hat{e}(s,0)$. We initialize the minimum cost label at the destination $\pi(q)$ to infinity. The algorithm stores the node-time pairs in the candidate set C as a priority queue where the smallest $\hat{\lambda}(i,t)$ label is selected. Initially, C includes only the source node s at time zero, i.e. $(s,0)$. The algorithm selects a node-time pair (i,t) with minimum label $\hat{\lambda}(i,t)$. If the label $\hat{\lambda}(i,t)$ is less than the minimum cost label $\pi(q)$ at the destination found so far, which means that the destination node q has not been visited yet (i.e. $i\neq q$), the arcs with positive residual capacity in the forward star and backward star (reverse arcs) of node i are explored (as before). Otherwise, if the selected node i is the destination node q, the algorithm is terminated since all node-time pairs in the forward and backward star of $(q,\tau(q))$ as well as the remaining node-time pairs in C cannot lead to a cost at the destination lower than $\pi(q)$.

3.4.3.2 Pseudocode of Algorithm C

Step 1: Computation of lower bounds

Obtain $\hat{e}(i,t)$ $\forall(i,t)\in N\times T$ (e.g. from DOT)

Step 2: Initialization

$\hat{\pi}(i,t)=\infty, \hat{\lambda}(i,t)=\infty, \forall(i,t)\in N\times T$; $\hat{\pi}(s,0)=0$; $\hat{\lambda}(s,0)=\hat{e}(s,0)$

$\pi(q)=\infty$; $C=\{(s,0)\}$

Step 3: Node selection

$$(i,t_i)=\underset{(j,t_j)\in C}{Arg\min}\,\hat{\lambda}(j,t_j)$$

$C = C \setminus \{(i,t_i)\}$

Step 4: Stopping criterion

If $i = q$, then stop. Otherwise, go to Step 5.

Step 5: Explore forward and backward star

For all $j \in A(i)$ such that $r_{ij}(t_i) > 0$ and $j \neq s$

$t_j = t_i + d_{ij}(t_i)$

If $t_j \leq M$ then

If $\left(\hat{\pi}(i,t_i) + c_{ij}(t_i) < \hat{\pi}(j,t_j)\right)$ and $\left(\hat{\pi}(i,t_i) + c_{ij}(t_i) + \hat{e}(j,t_j) < \pi(q)\right)$ then

$\hat{\pi}(j,t_j) = \hat{\pi}(i,t_i) + c_{ij}(t_i)$

$\hat{\lambda}(j,t_j) = \hat{\pi}(j,t_j) + \hat{e}(j,t_j)$

$p(j,t_j) = (i,t_i)$

If $(j,t_j) \notin C$, then $C = C \cup \{(j,t_j)\}$

If $j = q$, then $\pi(q) = \hat{\pi}(j,t_j)$

For all $j \in B(i)$ and $j \neq s$

For all t_j such that $t_i = t_j + d_{ji}(t_j)$ and $r_{ji}(t_j) < U_{ji}(t_j)$

If $\left(\hat{\pi}(i,t_i) - c_{ji}(t_j) < \hat{\pi}(j,t_j)\right)$ and $\left(\hat{\pi}(i,t_i) - c_{ji}(t_j) + \hat{e}(j,t_j) < \pi(q)\right)$ then

$\hat{\pi}(j,t_j) = \hat{\pi}(i,t_i) - c_{ji}(t_j)$

$\hat{\lambda}(j,t_j) = \hat{\pi}(j,t_j) + \hat{e}(j,t_j)$

$p(j,t_j) = (i,t_i)$

If $(j,t_j) \notin C$, then $C = C \cup \{(j,t_j)\}$

If $j = q$, then $\pi(q) = \hat{\pi}(j,t_j)$

Step 6: Check if candidate set is empty

If $C = \phi$, then stop; problem is infeasible. Otherwise, go to Step 3.

Note that although the pseudocode given above corresponds to Step 2 of the generic algorithm, the computation of lower bounds (Step 1 in the pseudocode) is not repeated for every augmentation. Moreover, for multiple minimum cost flow problems having the same destination, the computation of lower bounds would be done only once. Although it is not done in the current implementation, one can obtain tighter lower bounds on the travel costs. Since the minimum travel cost from any node-time pair to the destination is a non-decreasing function of the number of augmentations, one can update after every augmentation the lower bounds of those node-time pairs (i,t_i) that are on a shortest path (the augmenting path), as follows: $\hat{e}(i,t_i)=\pi(q)-\hat{\pi}(i,t_i)$, which is the difference between the minimum cost label at the destination and the minimum cost label at (i,t_i) (since $\hat{\pi}(i,t_i)=\pi(i,t_i)$ when the algorithm terminates) obtained in the given augmentation iteration.

3.4.3.3 Correctness of Algorithm C

Let π^* be the cost of a minimum cost path from $(s,0)$ to the destination q in the residual network. Before proving the correctness of Algorithm C, we state the following lemma.

Lemma 3.2: *Before Algorithm C terminates, there exists always a node-time pair* (i,t_i) *in the candidate set C such that: (1)* $\hat{\pi}(i,t_i)+\hat{e}(i,t_i)\le\pi^*$, *and (2)* (i,t_i) *is on a shortest path to* q.

The proof of Lemma 3.2 is given in Appendix B. ■

Corollary 3.3: *Every node-time pair* (j,t_j) *selected from C is such that:* $\hat{\pi}(j,t_j)+\hat{e}(j,t_j)\le\pi^*$.

The proof of Corollary 3.3 is given in Appendix B. ■

To argue the correctness of Algorithm C, we prove the following three properties:

(1) Algorithm C stops after a finite number of iterations.

Let Δ be the minimum cost of an arc. Any node-time pair (i,t_i) further than $K = \frac{\pi^*}{\Delta}$ arcs from $(s,0)$ is such that: $\hat{\lambda}(i,t_i) \geq \hat{\pi}(i,t_i) \geq \pi(i,t_i) > K.\Delta = \pi^*$. By Lemma 3.2, there is a node-time pair (j,t_j) in C such that $\hat{\lambda}(j,t_j) \leq \pi^*$. (j,t_j) would then be selected from C before (i,t_i). Therefore, any node-time pair further than K arcs from $(s,0)$ is never visited by Algorithm C.

Let $\mu(K)$ be the set of node-time pairs that are within K arcs from $(s,0)$. Failure of Algorithm C to terminate could then only be due to continued revisiting of node-time pairs in $\mu(K)$. Any node-time pair (i,t_i) in $\mu(K)$ is revisited at most a finite number of times $\omega(i,t_i)$ as there is a finite number of paths from $(s,0)$ to (i,t_i) passing only through nodes within K arcs from $(s,0)$. Let $\omega = \max_{(i,t_i)\in\mu(K)} \omega(i,t_i)$ denote the maximum number of times any node-time pair in $\mu(K)$ is revisited. Then, after at most $\omega|\mu(K)|$ selections, none of the node-time pairs in $\mu(K)$ will be revisited. Since node-time pairs outside $\mu(K)$ are not visited, Algorithm C must terminate.

(2) If the problem is feasible (i.e. there exists at least one path connecting $(s,0)$ to q), Algorithm C stops when the destination is selected (Step 4 of the algorithm).

Assume that the problem possesses a non-empty feasible domain. We prove by contradiction that the algorithm cannot exit at Step 6 (i.e. when the candidate set is empty). By Lemma 3.2, the last node-time pair (i,t_i) selected from C (where $i \neq q$) before C becomes empty must be on a shortest path P to the destination. Since there is at least one more node-time pair after (i,t_i) on P, one can invoke the same argument as in the proof of Lemma 3.2 to conclude that one of those node-time pairs after (i,t_i) on P must be in C after (i,t_i) is selected from C. Therefore, (i,t_i) cannot be the last node-time pair to be selected from C. Hence, the destination node will eventually be added to C, and the algorithm terminates when the destination node is selected (Step 4 of the algorithm).

(3) When Algorithm C terminates, the cost label at the destination is optimal.

By Corollary 3.3, when the destination node (q,t_q) is selected from C, $\hat{\lambda}(q,t_q)=\hat{\pi}(q,t_q)+\hat{e}(q,t_q)\le\pi^*$. But $\hat{e}(q,t_q)=0$. Therefore, $\hat{\pi}(q,t_q)\le\pi^*$, and the cost label $\hat{\pi}(q,t_q)$ is optimal. ■

3.4.3.4 Implementation Details

The implementation of Algorithm C corresponding to the numerical results of this chapter uses a binary heap data structure to implement the list of candidate nodes C. Other priority queue data structures can be used. Let N_A and N_S be respectively the number of nodes added to and selected from the heap during the course of Algorithm C. At most N_A nodes are stored in the heap at any one time. The number of selected nodes is less than or equal to the number of added nodes $(N_S \le N_A)$ because the algorithm can be terminated before all nodes in the heap are visited. The nodes are organized by estimated minimum cost labels to the destination from each node.

1. The initialization of the labels $\hat{\lambda}(i,t)$ takes $O(nM)$.
2. Selecting and removing the minimum element from the heap is done N_S times, and each time it takes $\theta(\log N_A)$.
3. Adding an element to the heap is done N_A times, and each time it takes $\theta(\log N_A)$.
4. Assuming that the average number of outgoing arcs from every node is equal to $\frac{m}{n}$, exploring the forward star and backward star of all selected nodes is done at most $\frac{2m}{n} * N_S$ times (counting also the reverse arcs). Exploring an arc can be done in $O(1)$ but might lead to updating the label of a node that is already in the heap and consequently to a percolate operation, which can be done in $\theta(\log N_A)$.

Therefore, the theoretical running time complexity of the specialized version of Algorithm C is $\theta\left(nM + N_A \log N_A + N_S \frac{m}{n} \log N_A\right)$. Note that the computation of the lower bounds $\hat{e}(i,t)$ can be done in $O(nM+mM)$ through algorithm DOT and is done

only once. If a static shortest path algorithm is used to compute lower bounds, then the run time of this step is in $O((n+m)\log n)$. One can in principle establish upper bounds on the numbers of selections and additions from the heap. However, those upper bounds would be too loose to assess the efficiency of the algorithm. Experimental results (Section 3.6) are a better tool in this case.

3.5 Special Cases and Extensions

In this section we revisit the assumptions that were made in the model formulation, and show how to extend the solution algorithms that we have developed for the new cases of interest. We specifically address the following issues: (1) the case where link travel costs are equal to their travel times, (2) the waiting-is-allowed policy at nodes, and (3) the case of multiple sources, multiple destinations, and multiple departure times.

3.5.1 The Case of the Minimum Travel Time Problem

A particular case of the time-dependent minimum cost flow problem is the time-dependent minimum travel time flow problem, which is obtained by setting the link costs equal to their travel times. Therefore, Algorithms A, B, and C described above can be directly used as shortest path algorithms in the solution algorithm to the minimum travel time flow problem. The efficiency of the adaptation of Algorithm A to the minimum time problem will not be discussed as the algorithm was originally developed to solve shortest paths in the general minimum cost flow problem. Its adaptation to the minimum time problem would lead to a less efficient algorithm. However, for Algorithms B and C, one can develop specialized versions that determine shortest paths in the solution algorithm to the minimum travel time flow problem more efficiently than the original versions by exploiting properties of link travel times as cost functions to reduce the number of operations performed by these algorithms. Below we briefly describe these specialized algorithms and provide their running time complexities. For a detailed description of these algorithms, the reader is referred to Chabini and Abou Zeid (2002). Moreover, we describe another algorithm in the literature that was proposed to solve the time-dependent minimum travel time flow problem.

3.5.1.1 Algorithm B Specialized for the Minimum Travel Time Flow Problem

The computation of shortest paths involved in the solution of the time-dependent minimum travel time flow problem can be viewed as a connectivity problem between the source node and all nodes that correspond to the destination node in the time-space network. As demonstrated in Section 3.4, a node-time pair needs to be visited at most once in every minimum time path computation. This would lead to a significant reduction in the number of node additions and selections, as compared to the general minimum cost flow problem, and consequently, to a decrease in running time. The running time of the specialized version of Algorithm B is independent of data structures used which allow for node additions, updates, and selections in $O(1)$ run time, as the order of node additions and selections does not affect the number of times the reachable nodes are visited. Therefore, the maximum number of nodes which could be visited by the algorithm is equal to nM, which is the total number of nodes in the time-space network. For every node visited, all forward and reverse arcs which have positive residual capacity are explored only once. Thus, the maximum number of arc explorations is equal to $2mM$ (mM forward arcs and mM reverse arcs). Consequently, when the specialized version of Algorithm B is used to compute shortest paths, the theoretical running time complexity of the time-dependent minimum travel time flow algorithm is in $O((nM + mM)v)$.

Moreover, the specialized version of Algorithm B implemented using a time-bucket data structure can be terminated when the destination node is selected. To show this, we first provide a lemma whose proof is given in Appendix C.

Lemma 3.4: *For any augmenting path, the arrival time at the destination is greater than the arrival time at any intermediate node-time pair on the augmenting path.*

To exploit this property, one can visit nodes in increasing order of time, taking into consideration the existence of reverse arcs with negative travel times. The implementation of an increasing order of time algorithm can be done by means of a time-bucket data structure, as described previously. When the destination node is selected from a bucket B_t, there is no need to explore higher time buckets as there exist no reverse arcs emanating from any node-time pair in a higher time bucket (see the proof of Lemma 3.4), and the algorithm can be terminated. Therefore, the specialized version of Algorithm B

applied in an increasing order of time would result in significant computational time savings as compared to the initial version of Algorithm B where all reachable node-time pairs are visited.

3.5.1.2 Algorithm C Specialized for the Minimum Travel Time Flow Problem

As the computation of shortest paths involved in the minimum time flow problem is a connectivity problem between the source node at a certain departure time and the destination node corresponding to the earliest arrival time, one can reduce the number of node-time pairs visited by the specialized version of Algorithm B. This is the main idea behind the specialized version of Algorithm C, which tries to direct the search towards the destination node, by using lower bounds on the minimum travel times from node-time pairs to the destination node. In the specialized version of Algorithm C, any node-time pair cannot enter the heap more than once, as explained before. We next analyze the running time complexity of the algorithm.

As before, let N_A and N_S be respectively the number of nodes added to and selected from the heap during the course of the specialized version of Algorithm C. The same running time analysis that was given for Algorithm C applies here as well. Note that for the specialized version of Algorithm C, N_A is less than or equal to the total number of node-time pairs in the time-space network, i.e. $N_A \leq nM$, since a node-time pair is added to the candidate set at most once. In practice, the number of additions N_A to the heap is much lower than nM due to the lower bound property which reduces the search area in the network. The number of explored arcs is at most $2mM$ which is the maximum number of arcs in the residual time-space diagram. Note that an update operation does not lead in this case to the percolation of any node in the heap, and can thus be done in $O(1)$. Therefore, the theoretical running time complexity of the specialized version of Algorithm C is in $O(nM\log(nM)+mM)$. In practice, however, this upper bound is almost never reached because of the low number of node selections and additions. When the specialized version of Algorithm C is used to compute shortest paths, the theoretical running time complexity of the time-dependent minimum travel time flow algorithm is in $O((nM\log(nM)+mM)v)$ since at most v augmentations will be done.

The specialized versions of Algorithms B and C were implemented, and their computational efficiencies were assessed by using large-size capacitated dynamic networks. The computational results indicate that the specialized version of Algorithm C is more efficient than the specialized version of Algorithm B due to the lower bound property used in the former algorithm. For the test networks used, the successive shortest path algorithm employing the specialized version of Algorithm C achieved significant time savings, compared to that employing the specialized version of Algorithm B by up to a factor of 15. For more details, the reader is referred to Chabini and Abou Zeid (2002).

3.5.1.3 Other Algorithms

As noted above, the specialized versions of Algorithms B and C can be used to compute shortest paths in the solution algorithm to the time-dependent minimum travel time flow problem, leading to running time complexities of $O((nM+mM)v)$ and $O((nM\log(nM)+mM)v)$, respectively. Below we provide a brief description of an existing algorithm in the literature which can solve the time-dependent minimum travel time flow problem and show its running time complexity.

In Miller-Hooks and Stock Patterson (2002), an algorithm which can solve the time-dependent minimum travel time flow problem where unlimited waiting is allowed at all nodes is developed. As is the case of the earlier algorithms described in this chapter, the algorithm in Miller-Hooks and Stock Patterson (2002) is also a particular case of the generic flow augmentation algorithm. It differs from the other algorithms in its way of computing minimum time paths. The shortest path algorithm developed in Miller-Hooks and Stock Patterson (2002) uses a dynamic adaptation of a label correcting algorithm which performs at most $O(nm)$ steps. Each of these steps involves an evaluation of the minimum travel time $D_{ij}(t)$ (see Chabini (1998), $D_{ij}(t)=\min_{ubw(i,t)+t\geq s\geq t}(s-t+d_{ij}(s))$, where $ubw(i,t)$ is the maximum waiting time allowed at node i at departure time t) corresponding to an arc (i,j) and a departure time t, which can be done in $O(M)$ time. Therefore, each shortest path computation is done in $O(nmM)$. Since at most v augmentations will be done, the theoretical running time complexity of the overall

solution algorithm developed in Miller-Hooks and Stock Patterson (2002) is in $O((nmM)v)$.

Note that if unlimited waiting is allowed at nodes, the specialized versions of Algorithms B and C can still be used to compute minimum time paths with the same running complexities as in the no-waiting case. We discuss waiting policies in the next section.

3.5.2 Waiting Policies

We refer the reader to Chabini and Dean (1998) for a comprehensive discussion of waiting policies and solution algorithms for shortest path problems where waiting is allowed. Here we briefly summarize some concepts and properties of waiting policies that will be useful in extending Algorithms B and C to allow for the possibility of waiting at nodes. In order to fully characterize a waiting policy, one needs to specify the structure of three time-varying waiting attributes: the time window of allowed waiting, the waiting cost, and the waiting capacity of a node i at time t. By time window, we refer to upper and lower bounds on the amount of waiting allowed at node i at time t, denoted respectively as $ubw(i,t)$ and $lbw(i,t)$. The waiting cost can be general if it is a function of the amount of waiting that has already occurred, or it can be memoryless otherwise. Let $cw_i(t,\tau)$ denote the cost of waiting at node i for τ units of time, if waiting starts at time t. Waiting capacity, denoted as $U_i(t)$, controls the amount of flow units that can be held at node i at time t.

For simplicity, assume that the lower bound on waiting is zero. To express the presence of a bounded waiting policy in the time-space network, one needs to add for every node time pair (i,t) vertical arcs connecting (i,t) to $(i,t+\tau)$, where $0 \le \tau \le ubw(i,t)$. The cost of every such arc is given by $cw_i(t,\tau)$, its travel time is equal to τ, and its capacity is equal to $U_i(t)$. This transformation of the time-space network results in the addition of $O(nM^2)$ arcs.

We discuss a waiting structure for which the specialized versions of Algorithms B and C can be used to solve the problem efficiently. Consider the minimum travel time

flow problem with unlimited waiting allowed at nodes and infinite waiting capacity. If unlimited waiting is allowed at nodes, the specialized versions of Algorithms B and C can be used with the same running time complexities as in the no-waiting case. To see this, note that unlimited waiting can be represented by adding for every node-time pair (i,t) in the time-space network a vertical arc from (i,t) to $(i,t+1)$. This transformation results in nM additional arcs in the time-space network, and thus the total number of arcs in the time-space network is $nM+mM$ instead of mM. Therefore, the running time complexities of the time-dependent minimum travel time flow algorithms with the specialized versions of Algorithms B and C used to compute shortest paths and with unlimited waiting allowed at nodes are still in $O((nM+mM)v)$ and $O((nM\log(nM)+mM)v)$, respectively.

3.5.3 Multiple Sources, Destinations, and Departure Times

In this section, we describe how Algorithms B and C (or their specialized versions) can be used as minimum cost (time) path algorithms in the successive shortest path algorithm in a network with multiple supply nodes and/or multiple demand nodes. Examples of network flow problems involving multiple sources and/or sinks are the evacuation (multiple destinations) and quickest transshipment problems (multiple sources and destinations). In the case of multiple sources, we also allow for multiple departure times.

One technique to solve these problems is to transform the given network and supply/demand structure into an equivalent network with one source and one destination, and then apply the minimum cost flow algorithms developed in this chapter to the transformed network. To transform a network with multiple sources into one with an equivalent single source, we create a supersource node S. We connect S to every source-time pair (s,t) with positive supply $v(s,t)$ by an arc that has a travel time equal to t, a travel cost equal to zero , and a capacity equal to $v(s,t)$ at time zero and equal to zero at all other times. Note that this transformation preserves the supply structure of the original problem as it ensures that the right supplies are available at the corresponding sources at the right departure times.

To transform a network with multiple destinations into one with an equivalent single destination, we similarly create an artificial superdestination node Q. In addition, we create a copy q_i' of every destination node q_i with positive demand $v(q_i)$. We connect q_i to q_i' by an arc that has zero travel time, zero travel cost, and infinite capacity. Additionally, we connect q_i' to Q by an arc that has zero travel time, zero travel cost, and capacity equal to $v(q_i)$ at time M and equal to zero at all other times. We allow for waiting at q_i' without any penalty. This transformation ensures that the amount of flow that departs q_i' at time M to the superdestination Q is exactly equal to the demand $v(q_i)$ of destination node q_i.

3.6 Computer Implementations and Numerical Results

We have implemented the various solution algorithms discussed in this chapter for the purpose of experimental testing. The objectives of the experimental study were the following: (1) analyze the running times of the solution algorithms for the dynamic minimum cost flow problem, where Algorithms A, B, and C are used to compute shortest paths, as a function of the following input parameters: the size of the network, the number of nodes, the number of arcs, the number of time periods, and the amount of flow that should be sent from the origin to the destination, (2) analyze the number of node-time pairs in the time-space network that are visited per augmentation by Algorithms B and C as a function of the size of the network, and (3) assess the practical computational performance and the time savings of Algorithm C as compared to Algorithms A and B.

3.6.1 Computer Implementations

We have developed computer implementations for Algorithms A, B and C. For Algorithm B, we have tested three implementations corresponding to three data structures: a time-bucket (as described previously), a dequeue, and a queue.

All the algorithms are coded in C++. The codes are available upon request. The tests were performed on a DELL Pentium III 933 megahertz computer with 256 megabytes of

RAM. The running times of the algorithms are reported in seconds, and they represent the average running time over 10 trials of each algorithm, where each trial corresponds to a different origin-destination pair.

3.6.2 Test Networks

Test networks were generated using a pseudo random network generator. Input to this network generator consists of: number of nodes, number of arcs, number of time intervals, range of link travel times, range of link travel costs, and range of link capacities. The topology networks are generated in two stages. First a cycle involving all nodes is created to ensure strong connectivity. Then the remaining number of links is added randomly.

3.6.3 Results

Table 3.1 shows the running times of the successive shortest path algorithm, using Algorithms A, B, and C to compute shortest paths, as a function of the size of the network, the number of nodes, the number of arcs, the number of time periods, and the amount of flow that should be sent from the origin to the destination. The ratios of the running times of the three algorithms to that where Algorithm C is used are reported in parentheses. For the test networks used, the solution algorithm employing Algorithm C to compute shortest paths achieved significant time savings compared to the other algorithms. The successive shortest path algorithm using Algorithm C was faster than that using Algorithm A by up to a factor of 113, and faster than that using Algorithm B by up to factors of 25, 39, and 72 for the time-bucket, dequeue, and queue implementations, respectively.

Table 3.2 (a) shows the number of node-time pairs N_A added to the candidate set C and the number of node-time pairs N_S selected from C, per augmentation, for Algorithm C as a function of network size. Table 3.2 (a) also shows the number of node-time pairs that are selected in Algorithm B for the time-bucket, dequeue, and queue implementations (In Algorithm B, the number of nodes added is equal to the number of nodes selected). In Algorithm A, all node-time pairs are visited, and so they are not

shown in the table. The computational time savings of the successive shortest path algorithm employing Algorithm C that will be reported below are due mainly to the small number of node-time pairs visited by Algorithm C. Table 3.2 (b) shows the average number of selections and additions (in %) per augmentation relative to the total number of node-time pairs. The results indicate that in Algorithm B a considerable part of the time-space network is explored. Moreover, as expected, different implementations of Algorithm B lead to different number of node visits. The results indicate that fewer nodes are visited using the time-bucket data structure than the dequeue and the queue data structures. Moreover, fewer nodes are visited using the dequeue than the queue. The effect of the number of node visits on running times is illustrated in Figures 3.3-3.7.

Figure 3.1 shows the variation in running times of the algorithms as a function of network size, with the number of arcs being three times the number of nodes. Figure 3.2 shows the running times as a function of the number of nodes. The number of arcs is held constant at 10000. Figure 3.3 shows the running times as a function of the number of arcs. The number of nodes is held constant at 100. Figure 3.4 shows the running times as a function of the number of time intervals. Finally, Figure 3.5 shows the running times as a function of the amount of flow that should be sent from the origin to the destination.

Figures 3.1-3.5 indicate that the running times of the three algorithms increase as a function of all network parameters. As network size, number of nodes, number of arcs, or number of time intervals increases, the size of the time-space network also increases. Thus, more node-time pairs could be reached by the algorithms. As the demand of flow units at the destination increases, more augmentation procedures could be done, and therefore the running time increases. However, note that for the successive shortest path algorithm employing Algorithm C, the marginal rate of increase in running time is small. The increase can be attributed to the fact that most of the work done in this algorithm is in the initialization phase and the computation of lower bounds, and these procedures are done only once irrespective of the amount of flow to be sent. The successive shortest path computations and augmentation procedures are very fast in comparison. In the solution algorithms where Algorithms A and B are used to compute shortest paths, the computation of shortest paths is the most time-consuming part of the algorithms, as in

each shortest path computation all nodes are visited in Algorithm A and all reachable nodes are visited in Algorithm B.

Based on above numerical results, it is seen that Algorithm C is more efficient than both Algorithms A and B. Moreover, the solution algorithm based on Algorithm B yields lower running times than that based on Algorithm A.

Numerical results that measure the effectiveness of the minimum travel time flow algorithm using the specialized versions of Algorithms B and C are reported in Chabini and Abou-Zeid (2002). The ratios of running times of the minimum cost flow algorithm to those of the minimum time flow algorithm were in the range of 2 and 1.2 for Algorithms B and C, respectively (and their specialized versions). As expected, the minimum cost flow algorithm has a higher run time than the minimum time flow algorithm because a node-time pair might be visited more than once in the shortest path algorithm (i.e. Algorithm B or C). However, for Algorithm B, the ratio is higher than for Algorithm C since in Algorithm C the most time-consuming part is the computation of lower bounds which is done only once, while in Algorithm B the time-consuming part is the exploration of the network.

Table 3.1. Running times (reported in seconds) of the successive shortest path algorithm employing Algorithms A, B, and C as a function of various network parameters. The ratios of running times of the three solution algorithms, with respect to that employing Algorithm C, are reported in parentheses. $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, and $U_{ij} \in [1,10]$.

m = 3n, T = 100, v = 20					
n	500	1000	2000	3000	4000
Alg. A	2.678	6.324	18.507	36.380	41.614
	(56.5)	(67.3)	(87.3)	(113.8)	(95.3)
Alg. B (Queue)	1.770	4.929	12.767	19.909	27.074
	(37.4)	(52.4)	(60.2)	(62.3)	(62.0)
Alg. B (Dequeue)	0.847	2.497	6.887	11.624	16.368
	(17.9)	(26.6)	(32.5)	(36.3)	(37.5)
Alg. B (Time-Bucket)	0.463	1.116	3.375	6.934	10.879
	(9.8)	(11.9)	(15.9)	(21.7)	(24.9)
Alg. C	0.047	0.094	0.212	0.320	0.437
	(1)	(1)	(1)	(1)	(1)

m = 10000, T = 100, v = 20				
n	1000	2000	3000	4000
Alg. A	23.487	30.915	34.788	34.262
	(86.9)	(109.5)	(108.7)	(86.8)
Alg. B (Queue)	5.554	14.583	20.578	23.219
	(20.6)	(51.7)	(64.3)	(58.8)

Alg. B (Dequeue)	3.758	9.081	12.351	13.581
	(13.9)	(32.2)	(38.6)	(34.4)
Alg. B (Time-Bucket)	2.834	5.260	7.643	8.463
	(10.5)	(18.6)	(23.9)	(21.4)
Alg. C	0.270	0.282	0.320	0.395
	(1)	(1)	(1)	(1)

n = 100, T = 100, v = 20

m	500	1000	2000	3000	4000	5000
Alg. A	0.549	1.053	1.791	2.991	3.886	4.859
	(46.2)	(47.7)	(44.4)	(50.2)	(50.6)	(50.0)
Alg. B (Queue)	0.150	0.207	0.256	0.350	0.415	0.472
	(12.7)	(9.4)	(6.4)	(5.9)	(5.4)	(4.9)
Alg. B (Dequeue)	0.101	0.175	0.242	0.350	0.416	0.479
	(8.5)	(7.9)	(6.0)	(5.9)	(5.4)	(4.9)
Alg. B (Time-Bucket)	0.092	0.182	0.256	0.376	0.455	0.495
	(7.8)	(8.2)	(6.3)	(6.3)	(5.9)	(5.1)
Alg. C	0.012	0.022	0.040	0.060	0.077	0.097
	(1)	(1)	(1)	(1)	(1)	(1)

n = 1000, m = 3000, v = 20

T	30	60	90	100	150	200
Alg. A	1.770	3.636	6.334	6.294	10.936	14.264
	(67.8)	(60.9)	(74.1)	(66.7)	(79.9)	(79.2)
Alg. B (Queue)	0.412	1.852	3.984	4.897	8.773	14.379
	(15.8)	(31.0)	(46.6)	(51.9)	(64.1)	(79.8)
Alg. B (Dequeue)	0.341	1.314	2.307	2.569	3.898	5.289
	(13.1)	(22.0)	(27.0)	(27.2)	(28.5)	(29.4)
Alg. B (Time-Bucket)	0.215	0.568	0.968	1.104	1.712	2.397
	(8.2)	(9.5)	(11.3)	(11.7)	(12.5)	(13.3)
Alg. C	0.026	0.060	0.085	0.094	0.137	0.180
	(1)	(1)	(1)	(1)	(1)	(1)

n = 1000, m = 3000, T = 100

v	1	5	10	15	20	25
Alg. A	0.583	1.610	2.679	4.557	6.294	9.012
	(6.7)	(18.0)	(29.5)	(49.1)	(66.7)	(94.0)
Alg. B (Queue)	0.383	1.280	2.172	3.650	4.897	6.860
	(4.4)	(14.3)	(23.9)	(39.3)	(51.9)	(71.6)
Alg. B (Dequeue)	0.232	0.728	1.208	1.994	2.569	3.663
	(2.7)	(8.2)	(13.3)	(21.5)	(27.2)	(38.2)
Alg. B (Time-Bucket)	0.116	0.318	0.515	0.833	1.104	1.516
	(1.3)	(3.6)	(5.7)	(9.0)	(11.7)	(15.8)
Alg. C	0.087	0.089	0.091	0.093	0.094	0.096
	(1)	(1)	(1)	(1)	(1)	(1)

Table 3.2. (a) Number of node-time pairs selected and added in Algorithms B and C per augmentation. (b) Average number of selections and additions (in %) made in Algorithms B and C per augmentation relative to the total number of node-time pairs as a function of network size. In (a) and (b), the number of arcs is three times the number of nodes, the number of time intervals is 100, the flow that should be sent is 20 units, $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, and $U_{ij} \in [1,10]$.

(a)

Number of Nodes	500	1000	2000	3000	4000
Number of Node Selections N_S Made in Alg. C	20.39	25.28	28.75	34.64	31.26
Number of Node Additions N_A Made in Alg. C	63.31	79.95	95.22	112.29	99.45
Number of Node Selections Made in Alg. B (Time-Bucket)	38723.01	76727.28	152885.50	223721.80	304651.50
Number of Node Selections Made in Alg. B (Dequeue)	43068.06	83892.27	173449.10	252432.60	339040.90
Number of Node Selections Made in Alg. B (Queue)	73956.72	148223.20	317176.90	431749.80	563987.50

(b)

Number of Nodes	500	1000	2000	3000	4000
Avg. Number of Selections Made in Alg. C Relative to Total Number of Node-Time Pairs	0.041 %	0.025 %	0.014 %	0.012 %	0.008 %
Avg. Number of Additions Made in Alg. C Relative to Total Number of Node-Time Pairs	0.127 %	0.080 %	0.048 %	0.037 %	0.025 %
Avg. Number of Selections Made in Alg. B (Time-Bucket) Relative to Total Number of Node-Time Pairs	77.446 %	76.727 %	76.443 %	74.574 %	76.163 %
Avg. Number of Selections Made in Alg. B (Dequeue) Relative to Total Number of Node-Time Pairs	86.136 %	83.892 %	86.725 %	84.144 %	84.760 %
Avg. Number of Selections Made in Alg. B (Queue) Relative to Total Number of Node-Time Pairs	147.913 %	148.223 %	158.588 %	143.917 %	140.997 %

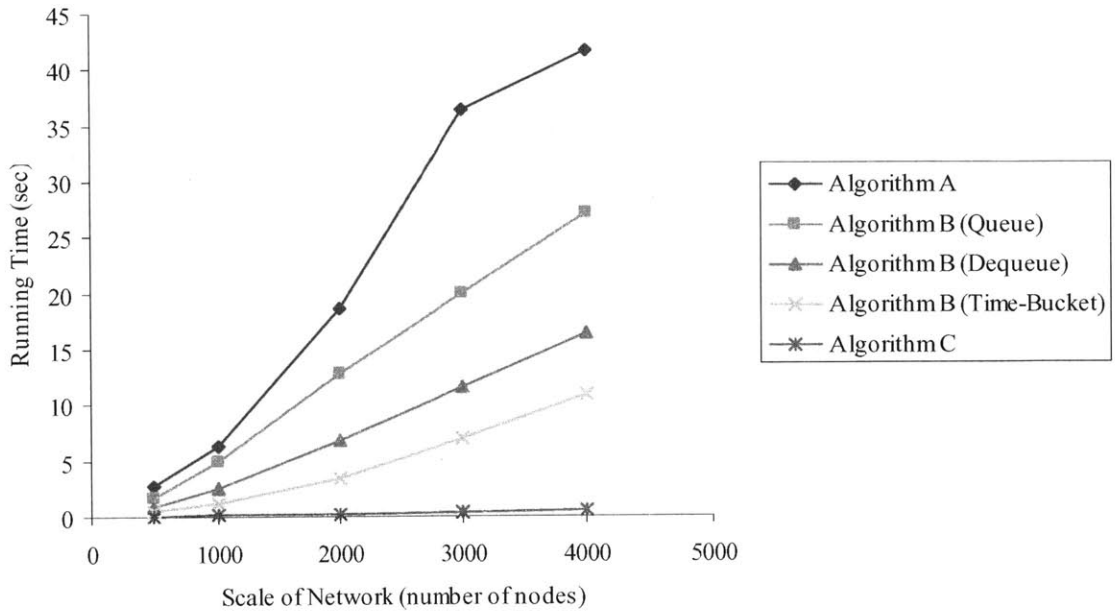


Figure 3.1. Running times of the successive shortest path algorithm employing Algorithms A, B, and C as a function of network size. The number of arcs is three times the number of nodes. The number of time intervals is 100. The flow that should be sent is 20 units. $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, $U_{ij} \in [1,10]$.

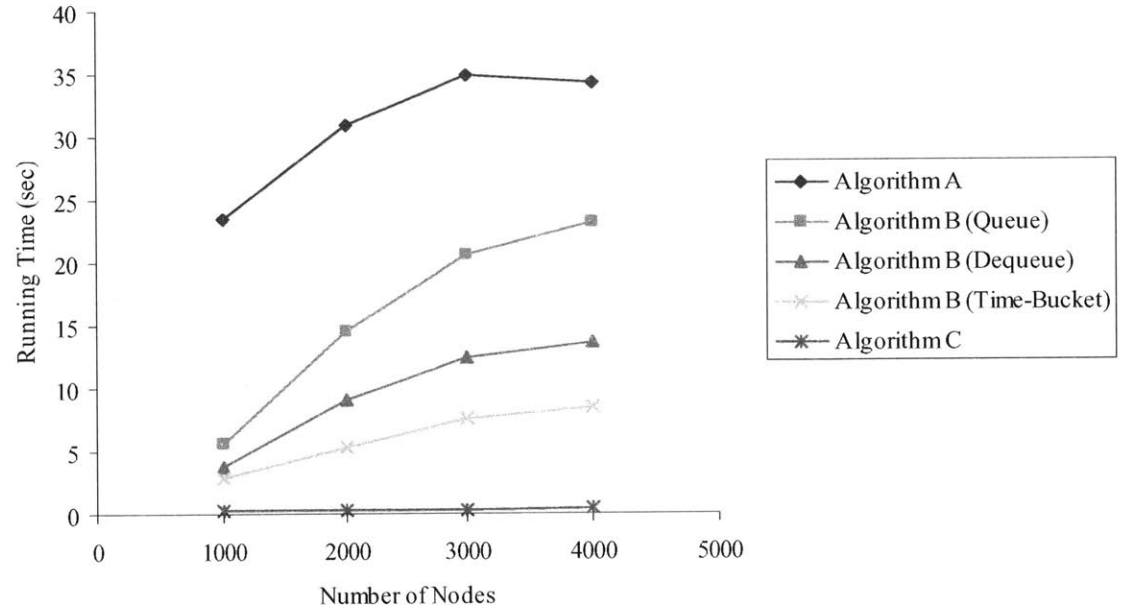


Figure 3.2. Running times of the successive shortest path algorithm employing Algorithms A, B, and C as a function of the number of nodes in the network. The number of arcs is 10000. The number of time intervals is 100. The flow that should be sent is 20 units. $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, $U_{ij} \in [1,10]$.

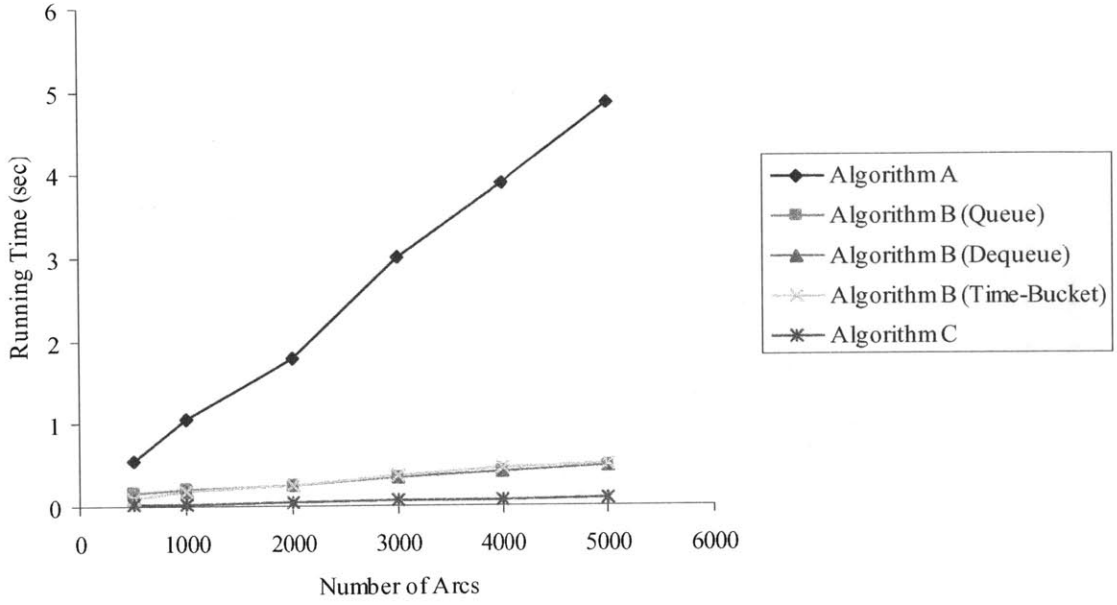


Figure 3.3. Running times of the successive shortest path algorithm employing Algorithms A, B, and C as a function of the number of arcs in the network. The number of nodes is 100. The number of time intervals is 100. The flow that should be sent is 20 units. $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, $U_{ij} \in [1,10]$.

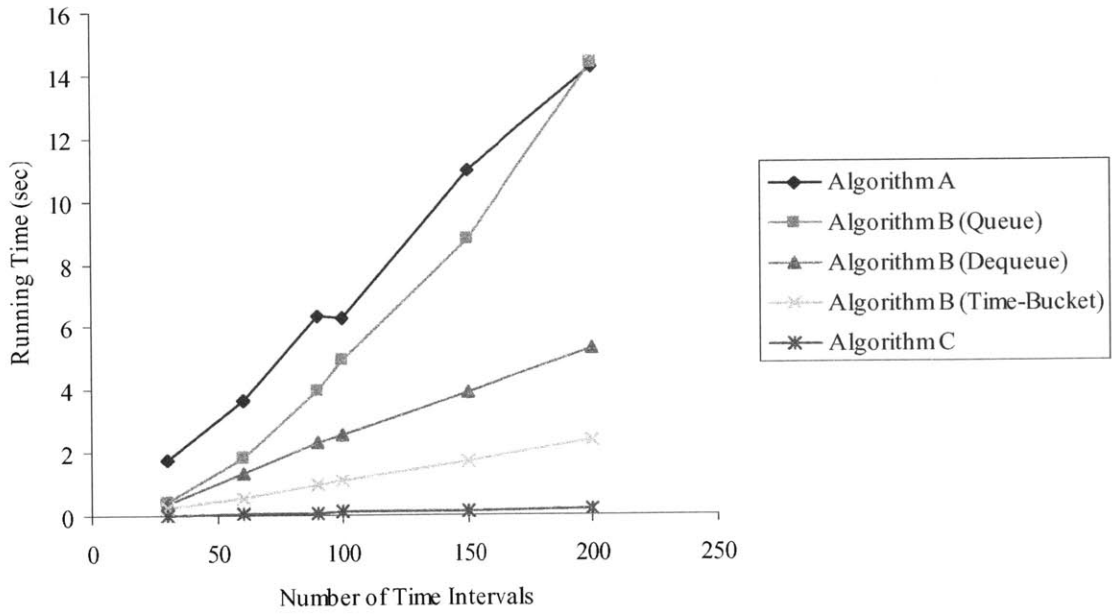


Figure 3.4. Running times of the successive shortest path algorithm employing Algorithms A, B, and C as a function of the number of time intervals. The number of nodes is 1000. The number of arcs is 3000. The flow that should be sent is 20 units. $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, $U_{ij} \in [1,10]$.

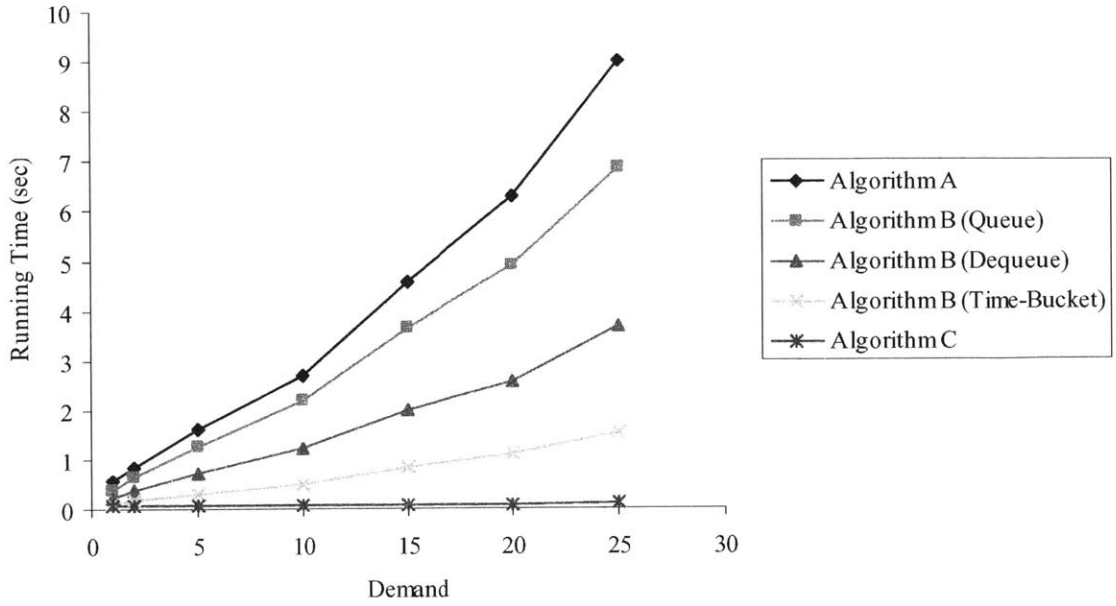


Figure 3.5. Running times of the successive shortest path algorithm employing Algorithms A, B, and C as a function of the demand of flow units at the destination. The number of nodes is 1000. The number of arcs is 3000. The number of time intervals is 100. $d_{ij} \in [1,5]$, $c_{ij} \in [1,7]$, $U_{ij} \in [1,10]$.

Chapter 4

Experimental Results Using Improved Shortest Path Algorithms

4.1 Introduction

The study of shortest path problems in static networks has been the subject of extensive research and has resulted in the development of various solution algorithms with different running time complexities (see Ahuja et al. (1993)). More recently, there has been an interest in developing shortest path algorithms for dynamic (time-dependent) networks where network attributes such as link costs and travel times depend on the entry time of the link (Grier and Chabini (2002), Chabini and Lan (2002), Pallottino and Scutellà (1998), Chabini (1998), Cai et al. (1997), Ziliaskopoulos (1994), Kaufman and Smith (1993), Orda and Rom (1990), Dreyfus (1969), and Cooke and Halsey (1966)). Shortest path problems arise in various application settings and often occur as subproblems to solve more evolved problems such as in the solution of dynamic traffic assignment models. Substantial research efforts have been directed at speeding up shortest path algorithms to meet the computational requirements of large-scale dynamic network flow models.

The objectives and contributions of this chapter are twofold. First we introduce a new algorithm (Algorithm Lazy-Sorting Label-Setting (LSLS)) developed in Chabini (2002)

for finding a single source shortest path tree in static networks. The new approach is different from traditional comparison-based label-setting shortest path algorithms as it tries to avoid finding a node with the minimum label at every iteration as the sole mean to label set a node. In Algorithm LSLS, certain optimality conditions defined on the node labels are utilized to determine if the labels are permanently set thus trying to avoid the need for sorting, which is the bottleneck operation in comparison-based label-setting algorithms. Second we extend the approach to solve one-to-all shortest paths in dynamic (time-dependent) First-In-First-Out (FIFO) networks for all departure times. We use the minimum arrival time labels corresponding to a certain departure time from the origin to reoptimize the shortest path computation corresponding to the following departure time. The optimality conditions we define for this problem also allow us to reduce the effort spent in sorting the nodes. We perform several experimental tests on large-scale hypothetical networks to assess the effectiveness of the new approaches in reducing the need for sorting and the resulting running times.

The remainder of this chapter is organized as follows. In Section 4.2, we describe the LSLS shortest path algorithm for static networks, provide its pseudocode, and conduct experimental tests to evaluate its effectiveness. In Section 4.3, we describe an application of the LSLS algorithm for dynamic FIFO networks, provide its pseudocode, and assess its effectiveness by testing it on several networks with varying parameters.

4.2 A New Approach for Solving the Shortest Path Problem in Static Networks

4.2.1 Description

In this section, we present Algorithm Lazy-Sorting Label-Setting (LSLS) developed in Chabini (2002) for computing shortest paths in static networks from one origin node to all other nodes. Algorithm LSLS is similar in structure to comparison-based label-setting algorithms, such as Dijkstra's algorithm, with modifications in the comparison-based steps. Let N denote the set of nodes and let A denote the set of arcs. Set S denotes the set of selected nodes, and set C denotes the set of candidate nodes (i.e. those that have

not been selected yet). The cost of an arc (i,j) is denoted as c_{ij}. We assume that arc costs are non-negative. Sets $A(i)=\{j:(i,j)\in A\}$ and $B(i)=\{j:(j,i)\in A\}$ represent the forward and backward star of node i, respectively. Let $\pi(i)$ be an upper bound on the minimum travel cost from the origin node s to node i. When the algorithm terminates, $\pi(i)$ is the minimum travel cost from s to i. The predecessor indices denoted as $\text{pred}(i)$ are used to trace a shortest path tree.

In Figure 4.1, we state Dijkstra's algorithm for clarity of presentation.

$S=\phi$; $C=N$;

$\pi(i)=\infty \quad \forall i\in N\setminus\{s\}$; $\pi(s)=0$; $\text{pred}(s)=0$.

While $|S|<|N|$

$i=\underset{j\in C}{Arg\min}(\pi(j))$;

$S=S\cup\{i\}$; $C=C\setminus\{i\}$;

For all $j\in A(i)$

If $\pi(j)>\pi(i)+c_{ij}$, then $\pi(j)=\pi(i)+c_{ij}$ and $\text{pred}(j)=i$.

Figure 4.1. Dijkstra's algorithm.

We note that the bottleneck operation in Dijkstra's algorithm is the determination of the node with the minimum cost label (select-min operation) since it requires sorting of the node labels and subsequent updating of data structures (e.g. priority queues). Different data structures have been suggested to develop implementations of Dijkstra's algorithm. They rely on reducing the time needed to perform the select-min operation (e.g. Dial's implementation, radix heap, etc.). Those implementations are also classified as comparison-based approaches since they employ some form of sorting to select the node with minimum cost label at every step.

It would be desirable to find a mechanism that can detect optimality of a cost label of a node, and select it in $O(1)$ time, and to use this mechanism to reduce the number of

nodes that undergo sorting. This is the essential idea of the LSLS Algorithm. It tries to avoid sorting nodes as much as possible (thus the name Lazy-Sorting), and instead to determine if a node has been permanently set by performing certain optimality checks. Let $\alpha(j)$ denote a lower bound on the minimum cost label of node j. Lemma 4.1 defines an optimality criterion for the label of a node.

Lemma 4.1: *If the label $\pi(j)$ of a node j satisfies $\pi(j) \leq \min_{i \in B(j) \text{and } i \in C} (\alpha(i) + c_{ij})$, then $\pi(j)$ is optimal.*

Proof:

The proof is by contradiction. Assume that $\pi(j) \leq \min_{i \in B(j) \text{and } i \in C} (\alpha(i) + c_{ij})$ but that $\pi(j)$ is not optimal. This means that $\pi(j)$ would be updated at some point to a label $\pi'(j) = \pi(i) + c_{ij} < \pi(j)$ by a node $i \in C$, where $i \in B(j)$. But $\pi(i) \geq \alpha(i)$ by the definition of $\alpha(i)$. Therefore, we have:

$\pi'(j) = \pi(i) + c_{ij} \geq \alpha(i) + c_{ij} \geq \min_{i \in B(j) \text{and } i \in C} (\alpha(i) + c_{ij}) \geq \pi(j)$. This contradicts the assumption that $\pi'(j) < \pi(j)$, and therefore $\pi(j)$ is optimal. ■

We define the caliber $l(j)$ of a node j as the minimum cost of all arcs in the backward star of j. $l(j) = \min_{(i,j) \in A} c_{ij}$. If j has no incoming arcs, the caliber $l(j)$ is equal to $+\infty$. Let $\min(C)$ denote the minimum cost label in the candidate set C ($\min(C)$ is $+\infty$, if C is empty). Corollaries 4.2 and 4.3 are immediate consequences of Lemma 4.1.

Corollary 4.2: *If the label $\pi(j)$ of a node j satisfies $\pi(j) \leq \min(C) + l(j)$, then $\pi(j)$ is optimal.*

Proof:

First, we prove that $\min(C)$ is a lower bound on the optimal labels of all nodes in C. That is, we can set $\alpha(i) = \min(C) \; \forall i \in N$. To prove the above property, we first need to prove that if the current label $\pi(p)$ of a node p is equal to $\min(C)$, then $\pi(p)$ must be optimal. Any path q from origin node s to node p consists of three subpaths: the first subpath contains nodes in S, the second subpath links the last node in S, say l, on path

q to the first node in C, say k, on path q, and the last subpath links node k to node p (k might coincide with p). Let $c^q(k) = \pi(l) + c_{lk}$ be the cost of the subpath of q from s to k. Note that $c^q(k) \geq \min(C)$ since $\min(C)$ is the minimum cost label in C after all nodes in S have updated their forward star (and l is one of those nodes). Since the cost of the subpath from k to p is greater than or equal to zero, then the cost of the path q from s to p is greater than or equal to $\min(C)$, and thus $\min(C)$ is optimal.

Now we prove that $\min(C)$ is a lower bound on the optimal labels of all nodes in C. There should be a node p in C whose current label is $\min(C)$ (and thus its optimal label) and whose predecessor is a node in S. Consider any other node j in C. An optimal path q from origin node s to node j consists of three subpaths: the first subpath contains nodes in S, the second subpath links the last node in S, say l, on path q to the first node in C, say k, on path q, and the last subpath links node k to node j. Notice that the current label $\pi(k)$ of node k must be equal to its optimal label (i.e. to the cost of the subpath of q from s to k) since node l (which is the predecessor of node k on the optimal path to k) has already updated its forward star. There are two cases: (1) If node k coincides with node p, then the optimal label of k is equal to the optimal label of p (i.e. equal to $\min(C)$). Since the optimal label of j is greater than or equal to the optimal label of k (since arc costs are non-negative), then the optimal label of j is greater than or equal to $\min(C)$. (2) If node k is different from node p, then the current label of node k (which is equal to its optimal label) is greater than or equal to the current label of node p (which is equal to its optimal label $\min(C)$) because $\min(C)$ is by definition the minimum current label in C. Since the optimal label of j is greater than or equal to the optimal label of k, then the optimal label of j is also greater than or equal to $\min(C)$. Thus, we have established that $\min(C)$ is a lower bound on the optimal labels of all nodes in C.

Since $l(j) \le c_{ij} \ \forall (i,j) \in A$, by the definition of $l(j)$, we have: $\min(C) + l(j) \le \min_{i \in B(j) \text{and } i \in C} (\alpha(i) + c_{ij})$, and thus $\pi(j) \le \min_{i \in B(j) \text{and } i \in C} (\alpha(i) + c_{ij})$. By Lemma 4.1, $\pi(j)$ must be optimal. ■

Corollary 4.3: *If the label $\pi(j)$ of a node j satisfies $\pi(j) \le \min(C) + \min_{j \in B(i) and\ j \in C} \{l(i) + c_{ij}\}$ after being updated from all nodes $i \in B(j)$, then $\pi(j)$ is optimal.*

Proof:

Suppose that $\pi(j) \le \min(C) + \min_{j \in B(i) and\ j \in C} \{l(i) + c_{ij}\}$ after being updated from all nodes $i \in B(j)$. If $\pi(j)$ were not optimal, $\pi(j)$ would be updated at some later stage by a node $i \in C$, where $i \in B(j)$, whose current label $\pi(i)$ would be updated to $\pi'(i) < \pi(i)$. That is we would have: $\pi'(j) = \pi'(i) + c_{ij}$. However, we have: $\pi'(i) \ge \min(C) + l(i)$. That is, $\alpha(i) = \min(C) + l(i)$ would be a lower bound on the label of any node $i \in B(j)$ and $i \in C$ that gets updated. Since $\pi(j) \le \min_{j \in B(i) and\ j \in C} \{\min(C) + l(i) + c_{ij}\} = \min_{j \in B(i) and\ j \in C} \{\alpha(i) + c_{ij}\}$, $\pi(j)$ must be optimal by Lemma 4.1. ■

After giving the optimality conditions for the label of a node, we are now ready to describe the LSLS algorithm. As in Dijkstra's algorithm, the nodes in the LSLS algorithm are organized in two sets: selected nodes S (i.e. those nodes that have been visited by the algorithm and whose forward star has been updated) and candidate nodes C. Moreover, the candidate set C is further divided into four disjoint subsets: permanently labeled nodes denoted as P, i.e. those nodes whose labels are known to be optimal but that have not been selected yet, two sets of candidate nodes H (heap) and D (delayed) that have been reached but not yet known to be optimal (i.e. whose labels can potentially still be updated), and a set of non-reached nodes denoted as NR.

The algorithm works as follows. As long as the set P of permanently labeled nodes is non-empty, the algorithm selects a node i from P, updates the cost labels of non-permanent nodes j in the forward star of i, and inserts j in D if j has not been reached before. When the set P becomes empty, the algorithm examines the node i with

the minimum label $\pi(j)$ in H, and if $\pi(j)$ is less than the minimum cost label in D, it inserts i in S and updates the cost labels of nodes j in the forward star of i. If j has not been reached before (i.e. $j \in NR$), the algorithm checks whether the inequality $\pi(j) \leq \pi(i) + l(j)$ is valid, where $\pi(j)$ is the minimum cost label of j found so far by the algorithm. If the inequality holds, node j is permanently set. This is valid because of Corollary 4.2, and because $\pi(i) = \min(C)$. Node j is then inserted in P. Otherwise, it is inserted in D in an attempt to delay (with the hope of avoiding) its entry to the heap. After every selection from H, the algorithm checks if any nodes have been inserted in P in which case these nodes are visited next.

When both P and H are empty, or when P is empty and the minimum cost label in H is greater than the minimum cost label in D, the algorithm visits all nodes i in D. For each node i in D, the algorithm updates $\pi(i)$ by examining all nodes j in its backward star that belong to one of the sets P, H, or D, i.e. $\pi(i) = \min\left\{\pi(i), \min_{j \in B(i) \text{ and } j \in P \cup H \cup D} \left(\pi(j) + c_{ji}\right)\right\}$. Then the algorithm checks if the following inequality is valid: $\pi(i) \leq \min(D) + \min_{j \in B(i) \, and \, j \in C} \left\{l(j) + c_{ji}\right\}$, where $\min(D)$ denotes the minimum cost label in D. If the inequality holds, node i is permanently set by Corollary 4.3, and is inserted in P. Otherwise, it is inserted in H. Note that the condition $\pi(i) \leq \min(D) + \min_{j \in B(i) \, and \, j \in C} \left\{l(j) + c_{ji}\right\}$ is stronger than the condition $\pi(i) \leq \min(D) + l(i)$, since $\min_{j \in B(i) \, and \, j \in C} \left\{l(j) + c_{ji}\right\} \geq l(i)$. Note that the inequality is strict if $c_{ji} > 0$. The algorithm terminates when all the nodes have been placed in S.

As such, the LSLS algorithm tries to delay the placement in the heap H of those nodes that have not yet been determined to be permanently set. In doing so, some nodes might move directly from D to P, thus avoiding the sorting step that would have been necessary if the node were inserted in H. Thus, the aim is to keep the set H as small as possible during the course of the algorithm as operations on this set are the most time-consuming parts of the label-setting algorithm. If set H remains empty or contains a very small number of nodes that is not a function of the input, then the run time of the LSLS algorithm would be linear (in the order of $(n+m)$).

4.2.2 Pseudocode of the Lazy-Sorting Label-Setting Algorithm for Static Networks

Step 1: Initialization

$S = \phi$; $H = \{s\}$; $D = \phi$; $NR = N \setminus \{s\}$;
$\pi(i) = \infty \quad \forall i \in N \setminus \{s\}$; $\pi(s) = 0$; $\text{pred}(s) = 0$;
$\min(D) = \infty$.

Step 2: Computation of Caliber (can be done outside of the algorithm as a preprocessing step)

$$l(j) = \min_{(i,j) \in A} c_{ij} \quad \forall j \in N$$

Step 3: Stopping Criterion
If $S = N$, then stop. Otherwise, go to Step 4.

Step 4: Visiting P

While $P \neq \phi$
  Select i from P; $P = P \setminus \{i\}$; $S = S \cup \{i\}$;
  For all $j \in A(i)$ such that $j \notin S \cup P$
    If $\pi(j) > \pi(i) + c_{ij}$ then
      $\pi(j) = \pi(i) + c_{ij}$
      $\text{pred}(j) = i$
      If $j \in NR$, then $D = D \cup \{j\}$
      If $j \in D$ and $\min(D) > \pi(j)$, then $\min(D) = \pi(j)$

If $H = \phi$ or $\min(H) \geq \min(D)$, then go to Step 6. Otherwise, go to Step 5.

Step 5: Visiting H

Select i from H; $H = H \setminus \{i\}$; $S = S \cup \{i\}$;
For all $j \in A(i)$ such that $j \notin S \cup P$
    If $\pi(j) > \pi(i) + c_{ij}$ then
      $\pi(j) = \pi(i) + c_{ij}$
      $\text{pred}(j) = i$
      If $j \in NR$ then
        If $\pi(j) \leq \pi(i) + l(j)$, then $P = P \cup \{j\}$. Otherwise, $D = D \cup \{j\}$.

If $j \in D$ and $\min(D) > \pi(j)$, then $\min(D) = \pi(j)$.

Go to Step 3.

Step 6: Visiting D

While $D \neq \phi$

Select i from D; $D = D \setminus \{i\}$.

For all $j \in B(i)$ such that $j \in P \cup H \cup D$

If $\pi(i) > \pi(j) + c_{ji}$ then

$\pi(i) = \pi(j) + c_{ji}$

$\text{pred}(j) = i$

If $\pi(i) \leq \min(D) + \min_{j \in B(i) \text{ and } j \in C} \{l(j) + c_{ji}\}$, then $P = P \cup \{i\}$.

Otherwise, $H = H \cup \{i\}$.

$\min(D) = \infty$

Go to Step 3.

Note that when the algorithm visits a node $i \in P$ and updates the label $\pi(j)$ of a node $j \in A(i)$, the optimality condition $\pi(j) \leq \min(C) + l(j)$ is not checked in the current implementations. The rationale behind this is that $\pi(j) = \pi(i) + c_{ij}$, and since $c_{ij} \geq l(j)$ and $\pi(i) \geq \min(C)$ with a high chance that $\pi(i) > \min(C)$, then that there is a low chance that the inequality $\pi(j) = \pi(i) + c_{ij} \leq \min(C) + l(j)$ is satisfied. The reason that $\pi(i) > \min(C)$ holds with a high chance is that when the algorithm visits node i, $\min(C)$ is equal to the label of the predecessor node k of i if k were in the heap, and so $\pi(i) > \pi(k) = \min(C)$ if $c_{ki} > 0$. Otherwise, if i were inserted in P when the set D was emptied, then $\pi(i) \geq \min(D) = \min(C)$.

4.2.3 Experimental Results

We have implemented the LSLS algorithm described above to test its effectiveness in reducing the number of nodes that need to be sorted and its running time. All codes are written in C++. The tests were performed on a DELL Pentium III 933 megahertz

computer with 256 megabytes of RAM. The running times of the algorithms are reported in seconds, and they represent the average running time over 10 trials of each algorithm, where each trial corresponds to a different origin node. The random network generator is the same as that used in the computational experiments in Chapter 3 of this thesis.

In the following, we report numerical results using two implementations of the LSLS algorithm. Implementation 1 refers to the algorithm described above except that when the nodes in set D are visited, their cost labels are not updated by visiting nodes in their backward star and the optimality check performed on a node j in set D is: $\pi(j) \leq \min(D) + l(j)$. Implementation 2 refers to the pseudocode given above, where the cost labels of nodes in set D are updated by visiting nodes in their backward star and the optimality check performed on a node j in set D is: $\pi(i) \leq \min(D) + \min_{j \in B(i) \text{ and } j \in C} \{l(j) + c_{ji}\}$.

We report two measures of effectiveness: the running time of the LSLS algorithm as compared to that of Dijkstra, and the number of nodes that are known to be optimal and that need not be sorted (denoted as nodes in P). Tables 4.1, 4.2, and 4.3 show the results as a function of network size, number of nodes, and number of arcs, respectively. Figures 4.2, 4.3, and 4.4 compare the percentage of nodes that are known to be optimal using the optimality conditions for both implementation 1 and implementation 2 as a function of network size, number of nodes, and number of arcs, respectively. Link costs belong to the following range: $c_{ij} \in [1,3]$.

Comparing implementation 1 to implementation 2, the percentage of nodes that are determined to be optimal through the optimality conditions is significantly greater for implementation 2 than for implementation 1 as a function of all network parameters. This is expected since in implementation 2 the optimality checks on nodes in set D are stronger, as explained previously. It can be seen from the results that for implementation 2 more than 70 % of the nodes can be known to be optimal without the need to sort them. This suggests that one can almost avoid the costly sorting operations performed on priority queues by enforcing stronger optimality conditions.

Comparing the running times of both implementations of the LSLS algorithm to the running time of Dijkstra's algorithm, implementation 1 and Dijkstra's algorithm have

similar running times for the test networks used. However, the running time of implementation 2 is greater than that of implementation 1 and Dijkstra's algorithm due to the additional work performed in examining the backward star of nodes in D. The latter result is somehow surprising due to the fact that a substantial number of nodes are known to be optimal through the optimality checks. We interpret those running times as indicative of the current implementation of the algorithms but not as conclusive of the expected experimental performance. Enhanced implementations of the LSLS algorithm could result in more savings in running times, which is a topic for future research.

Finally, Figure 4.5 compares the number of nodes that are known to be optimal through the optimality conditions for different cost ranges as a function of network size and using implementation 2. It is observed that as the cost range increases, the number of nodes that get inserted in P decreases. This is intuitive since if the cost range is small and the label of a node j gets updated from a node i, then it is more likely that the caliber of node j would be equal to the cost of arc (i, j) than if the cost range were larger. Consequently, if the cost range is small, a larger proportion of the nodes would satisfy Lemma 4.2. In the extreme case where the cost is uniform among all links, all nodes would satisfy the optimality condition (see Figure 4.5).

Table 4.1. Summary of results as a function of network size for implementation 1 (part a) and implementation 2 (part b) of the Lazy-Sorting Label-Setting algorithm for static networks. The number of arcs is three times the number of nodes.

(a)

Number of Nodes	t (LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
1000	0.00044	0.00051	1.15	483.60	48.36
2000	0.00108	0.00151	1.39	913.30	45.67
3000	0.00186	0.00187	1.00	1400.90	46.70
4000	0.00295	0.00272	0.92	1830.10	45.75

(b)

Number of Nodes	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
1000	0.00051	0.00051	1.01	763.60	76.36
2000	0.00128	0.00151	1.18	1461.30	73.07
3000	0.00271	0.00187	0.69	2172.90	72.43
4000	0.00481	0.00272	0.57	2879.30	71.98

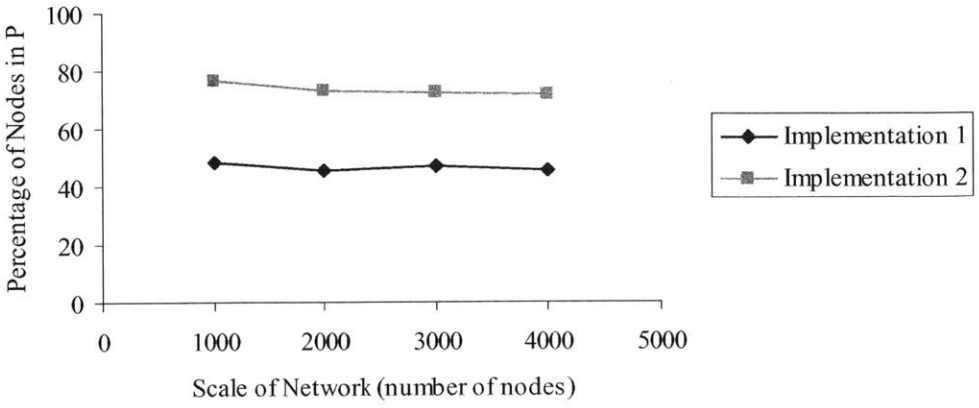


Figure 4.2. Percentage of nodes in P as a function of network size for implementation 1 and implementation 2 of the Lazy-Sorting Label-Setting algorithm for static networks. The number of arcs is three times the number of nodes.

Table 4.2. Summary of results as a function of the number of nodes for implementation 1 (part a) and implementation 2 (part b) of the Lazy-Sorting Label-Setting algorithm for static networks. The number of arcs is held constant at 15000.

(a)

Number of Nodes	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
1000	0.00086	0.00093	1.09	708.20	70.82
3000	0.0024	0.00231	0.96	1301.40	43.38
6000	0.00557	0.00435	0.78	2790.40	46.51
9000	0.01218	0.00703	0.58	4612.40	51.25
12000	0.01305	0.01163	0.89	5997.50	49.98

(b)

Number of Nodes	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
1000	0.00207	0.00093	0.45	985.80	98.58
3000	0.00416	0.00231	0.56	2117.30	70.58
6000	0.00854	0.00435	0.51	4479.80	74.66
9000	0.01376	0.00703	0.51	7429.90	82.55
12000	0.01732	0.01163	0.67	10752.30	89.60

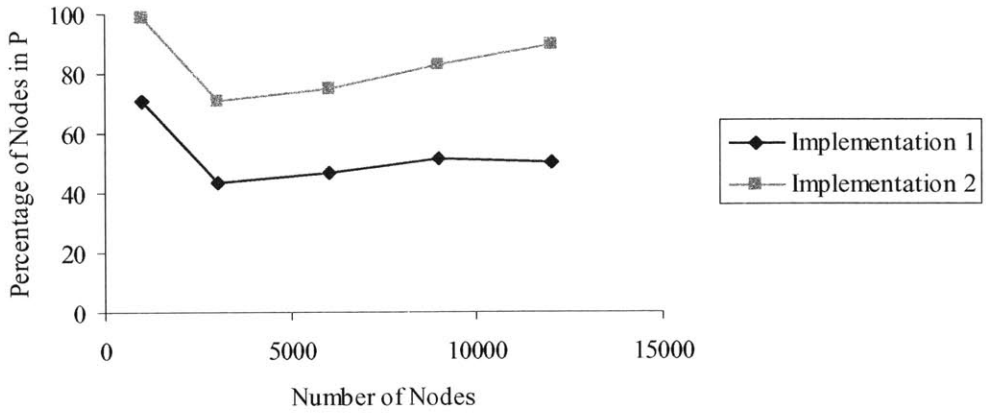


Figure 4.3. Percentage of nodes in P as a function of the number of nodes for implementation 1 and implementation 2 of the Lazy-Sorting Label-Setting algorithm. The number of arcs is held constant at 15000.

Table 4.3. Summary of results as a function of the number of arcs for implementation 1 (part a) and implementation 2 (part b) of the Lazy-Sorting Label-Setting algorithm for static networks. The number of nodes is held constant at 1000.

(a)

Number of Arcs	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
3000	0.00044	0.00051	1.16	499.80	49.98
6000	0.00053	0.00061	1.17	491.00	49.10
9000	0.00059	0.00067	1.14	589.60	58.96
12000	0.00073	0.00083	1.14	685.10	68.51
15000	0.0009	0.00096	1.06	699.70	69.97
50000	0.00328	0.00321	0.98	864.70	86.47
100000	0.00551	0.00690	1.25	936.50	93.65

(b)

Number of Arcs	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
3000	0.0005	0.00051	1.02	782.60	78.26
6000	0.0008	0.00061	0.76	785.50	78.55
9000	0.00117	0.00067	0.57	901.30	90.13
12000	0.00153	0.00083	0.54	968.20	96.82
15000	0.00201	0.00096	0.48	987.70	98.77
50000	0.0063	0.00321	0.51	999.00	99.90
100000	0.0108	0.00690	0.64	999.00	99.90

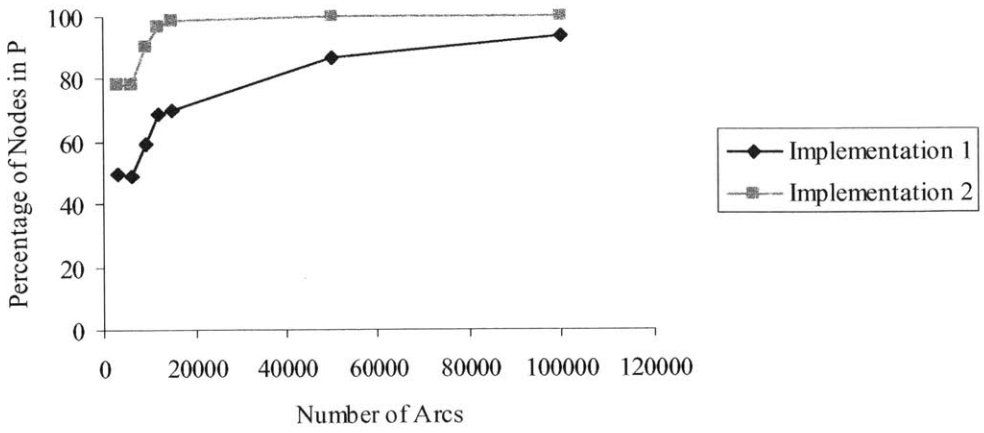


Figure 4.4. Percentage of nodes in P as a function of the number of arcs for implementation 1 and implementation 2 of the Lazy-Sorting Label-Setting algorithm for static networks. The number of nodes is held constant at 1000.

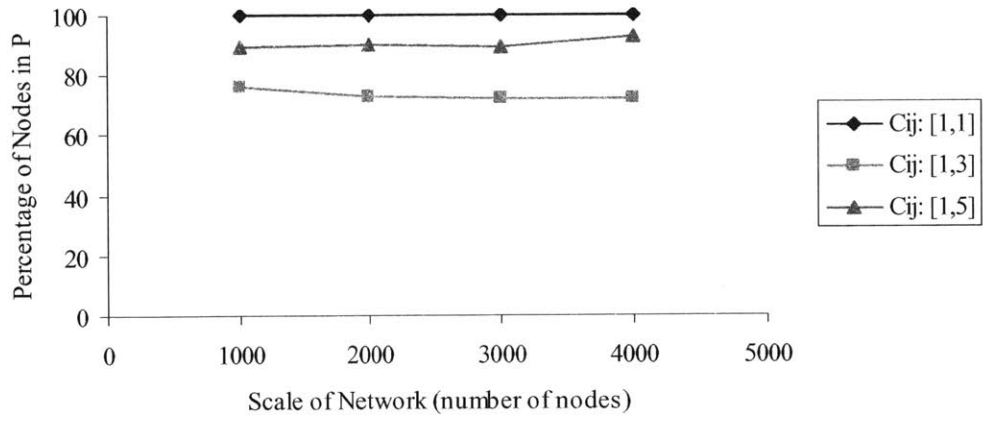


Figure 4.5. Percentage of nodes in P as a function of network size and cost range for implementation 2 of the Lazy-Sorting Label-Setting algorithm for static networks. The number of arcs is three times the number of nodes.

4.3 Application of the Lazy-Sorting Label-Setting Algorithm to Dynamic FIFO Networks

4.3.1 Description

In this section, we extend the LSLS algorithm presented in the previous section to solve shortest path problems in dynamic FIFO (First-In-First-Out) networks for the case of one origin to all destinations and all departure times. Note that if a dynamic network is FIFO, one can use Dijkstra's algorithm to compute the minimum arrival times at the nodes for a certain departure time (see Chabini (1998)). Similarly to Section 4.2, we will use optimality conditions to check if a node is already permanently set. However, the optimality conditions are different. Let $T = \{0,1,...,M\}$ denote the set of departure times. The FIFO property implies that one cannot arrive earlier at a path (or link) destination by departing its origin later. That is, the arrival time function is non-decreasing. Mathematically, this is equivalent to the following condition: $\pi_j(t) \leq \pi_j(t+1) \quad \forall t \in \{0,1,...,M-1\}$, where $\pi_j(t)$ denotes the minimum arrival time at node j if one departs origin node s along some path P at time t. Thus, the minimum arrival time at node j if one departs origin node s at time t is a lower bound on the minimum arrival time at node j if one departs origin node s at time $t+1$. One can then utilize the following optimality check to detect if a node's label is optimal.

Lemma 4.4: *If the arrival time label $\pi_j(t+1)$ of node j at time $t+1$ is equal to the minimum arrival time label $\pi_j(t)$ of node j at time t, then $\pi_j(t+1)$ is optimal.*

The data structures and selection rules used in the algorithm are the same as those used in the implementation of the LSLS algorithm for static networks. The only difference is in the optimality condition used to check if an arrival time label corresponding to a certain departure time is optimal. Below we give the pseudocode of the LSLS algorithm adapted for dynamic FIFO networks. We let $d_{ij}(t)$ denote the travel time on arc (i,j) if one departs node i at time t. Moreover, we assume that the network becomes static after a time horizon M, i.e. $d_{ij}(t) = d_{ij}(M) \quad \forall t > M$.

4.3.2 Pseudocode of the Lazy-Sorting Label-Setting Algorithm for Dynamic FIFO Networks

Step 1: First Departure Time

Use Dijkstra's algorithm or LSLS algorithm to find minimum arrival times for departure time zero. $t = 1$.For $t = 1$ to M, perform the following steps:

Step 2: Stopping Criterion for Whole Algorithm

If $t = M$, then stop. Otherwise, $t = t + 1$, and go to Step 3.

Step 3: Initialization

$S = \phi$; $H = \{s\}$; $D = \phi$; $NR = N \setminus \{s\}$;
$\pi_i(t) = \infty \quad \forall i \in N \setminus \{s\}$; $\pi(s) = t$; $\text{pred}(s) = 0$;
$\min(D) = \infty$.

Go to Step 4.

Step 4: Stopping Criterion for the Algorithm at a Certain Departure Time

If $S = N$, then go to Step 2. Otherwise, go to Step 5.

Step 5: Visiting P

While $P \neq \phi$
  Select i from P; $P = P \setminus \{i\}$; $S = S \cup \{i\}$;
  For all $j \in A(i)$ such that $j \notin S \cup P$
    If $\pi_j(t) > \pi_i(t) + d_{ij}(\pi_i(t))$ then
      $\pi_j(t) = \pi_i(t) + d_{ij}(\pi_i(t))$
      $\text{pred}(j) = i$
      If $j \in NR$ then
        If $\pi_j(t) = \pi_j(t-1)$, then $P = P \cup \{j\}$. Otherwise, $D = D \cup \{j\}$.
      If $j \in D$ and $\min(D) > \pi(j)$, then $\min(D) = \pi(j)$.

If $H = \phi$ or $\min(H) \geq \min(D)$, then go to Step 7. Otherwise, go to Step 6.

Step 6: Visiting H

Select i from H; $H = H \setminus \{i\}$; $S = S \cup \{i\}$;
For all $j \in A(i)$ such that $j \notin S \cup P$

If $\pi_j(t) > \pi_i(t) + d_{ij}(\pi_i(t))$ then

$\pi_j(t) = \pi_i(t) + d_{ij}(\pi_i(t))$

$\text{pred}(j) = i$

If $j \in NR$ then

If $\pi_j(t) = \pi_j(t-1)$, then $P = P \cup \{j\}$. Otherwise, $D = D \cup \{j\}$.

If $j \in D$ and $\min(D) > \pi(j)$, then $\min(D) = \pi(j)$.

Go to Step 4.

Step 7: Visiting D

While $D \neq \phi$

Select i from D; $D = D \setminus \{i\}$.

If $\pi_i(t) = \pi_i(t-1)$, then $P = P \cup \{i\}$. Otherwise, $H = H \cup \{i\}$.

$\min(D) = \infty$

Go to Step 4.

4.3.3 Experimental Results

We have implemented the algorithm described above using the C++ programming language to test its effectiveness in reducing the number of nodes that need to be sorted and its running time. The computer platform and network generator used are the same as those used in the computational tests described in Section 4.2.3.

Tables 4.4, 4.5, and 4.6 show the results as a function of network size, number of nodes, and number of arcs, respectively. Link travel times belong to the following range: $d_{ij}(t) \in [1,3] \quad \forall t \in T$. The set of departure times considered in these experiments is $T = 100$. The results indicate that the running times of the LSLS algorithm for dynamic FIFO networks and Dijkstra's algorithm are similar in magnitude. In the results, we also summarize the number of nodes that are known to be optimal through the FIFO optimality condition. The reported numbers represent the average number of nodes that are known to be optimal, where the average is computed over all departure times. As a function of network size (Table 4.4), the percentage of nodes that verify the optimality condition and avoid being placed in the heap is almost constant (around 25 %). As the

network becomes sparser (Table 4.5), the number of nodes that verify the optimality condition increases, while it decreases as the network becomes denser (Table 4.6). This might be due to the fact that as the network becomes denser, the probability of having strict FIFO arcs on a path increases. As a result, the probability of strict FIFO arrivals at the nodes increases, and the number of nodes satisfying Lemma 4.4 decreases.

Table 4.4. Summary of results as a function of network size for the Lazy-Sorting Label-Setting algorithm for FIFO dynamic networks. The number of arcs is three times the number of nodes.

Number of Nodes	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
1000	0.06263	0.0675389	1.08	253.30	25.33
2000	0.14081	0.1439591	1.02	493.70	24.69
3000	0.22799	0.2339348	1.03	801.10	26.70
4000	0.33854	0.3378199	1.00	1044.40	26.11

Table 4.5. Summary of results as a function of the number of nodes for the Lazy-Sorting Label-Setting algorithm for FIFO dynamic networks. The number of arcs is held constant at 15000.

Number of Nodes	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
1000	0.14238	0.13349	0.94	71.10	7.11
3000	0.29492	0.28886	0.98	518.10	17.27
6000	0.60861	0.529	0.87	1944.80	32.41
9000	0.99358	0.76459	0.77	3891.00	43.23
12000	1.35979	0.94785	0.70	6275.80	52.30

Table 4.6. Summary of results as a function of the number of arcs for the Lazy-Sorting Label-Setting algorithm for FIFO dynamic networks. The number of nodes is held constant at 1000.

m	t(LSLS)	t(Dijkstra)	t(Dijkstra)/t(LSLS)	Nodes in P	% of Nodes in P
3000	0.06263	0.06754	1.08	253.30	25.33
6000	0.08361	0.08205	0.98	142.40	14.24
9000	0.10104	0.1019	1.01	104.40	10.44
12000	0.1155	0.11496	1.00	90.00	9.00
15000	0.14238	0.13349	0.94	71.10	7.11
50000	0.38109	0.36039	0.95	34.50	3.45
100000	0.72881	0.68638	0.94	29.30	2.93

Chapter 5

Congestion and Emission Pricing in Dynamic Traffic Networks

5.1 Introduction

5.1.1 Background

Road pricing is a market-based policy instrument that has been advocated by economists as a means of revenue generation or traffic demand management. Despite a general attitude of public opposition on the basis of equity issues, recent interest in road pricing, particularly in a form known as congestion pricing, has been spurred by federal legislation (e.g. Intermodal Surface Transportation Efficiency Act of 1991 which authorized funding for planning and implementation of congestion pricing demonstration projects), and advances in road pricing technology (e.g. electronic road pricing) (Lo and Hickman (1997)). The rationale behind congestion pricing is that the average trip cost perceived by an individual does not capture the full external costs imposed on other users of the network. The problem is then to determine the prices that should be charged to travelers in order to account for the actual congestion costs induced by their use of the network. Congestion pricing is a type of responsive pricing that can change consumption patterns (Vickrey (1994)) by influencing users' travel choices at various levels: route choice, departure time choice, mode choice, destination choice, and frequency of travel.

Emission pricing, is another form of road pricing, that is aimed however at the pricing of the emission externality. It has not been studied as extensively as congestion pricing, but is receiving more attention lately as policy-makers strive to come up with economic-incentive approaches to combat pollution and meet air quality standards (Nagurney (2000 b)).

The framework that we propose in this chapter can be used to model pricing for congestion, emissions, or a combination of the two criteria. The solution algorithms that we develop can as well solve all these variants. Moreover, the basic framework can be extended to account for cases where pricing is done to optimize one criterion subject to constraints on the other criterion. For instance, pricing can aim at reducing congestion subject to an upper bound on the total emissions generated, or vice versa. The presentation, however, will be mainly focused on congestion pricing for simplicity and for its popularity as a pricing concept, but will be extended later in the chapter to cover the variants mentioned above.

5.1.2 Taxonomy

Congestion pricing has been studied in the literature from different modeling perspectives and under various assumptions, which we classify by referring to a taxonomy based on the following dimensions:

Theory: The theory of marginal cost pricing (see Appendix D for more details), dating back to Pigou (1920) and further explored by Walters (1961) and Vickrey (1969), postulates that in order to maximize the economic efficiency of trip-making, a toll equal to the difference between the marginal social cost and marginal (for the additional user) private cost should be levied. However, technical or political constraints could render the implementation of marginal cost pricing infeasible. Consequently, this has resulted in what is known as second-best pricing, i.e. the best that can be done given that marginal cost pricing cannot be employed (McDonald et al. (1999)), and has received considerable attention from the research community. Examples of second-best pricing include pricing only a subset of the links or paths in the network and/or imposing upper bounds on the prices.

Objectives: Congestion pricing ultimately aims at optimizing a certain system performance which varies with the assumptions made on travel demand. If the demand is inelastic to the prices, the objective function could be to minimize total travel time. In the case of elastic demand, minimizing total travel time can be achieved by setting the tolls large enough to drive the demand to zero, which is impractical in reality. In this case, therefore, the objective function could be to maximize net social benefit (defined as the difference between the total network user benefit from travel and the total social cost incurred by all network users).

Type of temporal analysis: This refers to whether travel demands, network conditions (such as travel times and capacities), and tolls are time-varying (dynamic) or not (static).

Pricing strategies: These include (1) link-based tolls, (2) path-based tolls, (3) origin-destination (O-D) based tolls, (4) destination-based tolls, (5) zone-based tolls (e.g. cordon pricing), (6) tolls for distance traveled and/or time spent in the network, and (7) vehicle-category-based tolls (mostly applicable in emission pricing). The reader is referred to Gomez-Ibanez and Small (1994) and Lo and Hickman (1997) for a more comprehensive review of pricing strategies.

User classes: Users can be classified according to one or more of the following criteria: travel time or cost perceptions, information access, value of time, and vehicle category. For instance, in congestion pricing, it is important to account for differences in valuations of time in analyzing the effects of a particular pricing scheme (e.g. commercial vehicles versus ordinary travelers). In emission pricing, the tolls can be made to vary by vehicle category to reflect differences in emission rates among vehicle categories.

Next we provide a review of the literature on congestion pricing as related to the above taxonomy.

5.1.3 Literature Review

In static traffic networks, after the pioneering works of Pigou (1920) and Knight (1924), marginal cost pricing has been developed in Walters (1961), Vickrey (1969), and Dafermos and Sparrow (1971). The classical two-route problem, where an untolled alternative road is available parallel to a tolled road, has been studied by Lévy-Lambert (1968), Marchand (1968), Braid (1996), and Verhoef et al. (1996). In Yang and Bell

(1997), an elastic demand model with queues is given and a bi-level programming approach is employed to select the best tolling policy that replaces queuing delays with an equivalent amount of tolls. Verhoef (2002) studies second-best pricing in static congested networks with perfect driver information and elastic demand. He derives a general solution for social welfare maximization based on a bi-level programming approach (Stackelberg game). Bi-level models of traffic management have also been studied in Patriksson and Rockafellar (2002), Clegg at al. (2001), Labbé et al. (1998), and Brotcorne et al. (2001). Recent developments (Liu and McDonald (1998, 1999), Small (1992 b), and Liu and Boyce (2002)) address multiple-period pricing models.

Dynamic congestion pricing models, where network conditions and link tolls are time-varying, have been addressed in Henderson (1974), Agneiv (1977), Arnott et al. (1990), Carey and Srinivasan (1993), and Huang and Yang (1996). Moreover, in Arnott et al. (1990), the effectiveness of various pricing policies (time-varying, uniform, and step tolls) are compared. The limitations of those models are that they consider either a bottleneck or a single-destination network. Wie and Tobin (1998) develop dynamic marginal cost pricing models for general transportation networks. As indicated by the authors, the application of their model is limited to destination-specific (rather than link or route based) tolling strategies, which might not be easy to implement in practice. Moreover, since the tolls are based on marginal cost pricing, it is implicitly assumed that all links can be priced. In Viti et al. (2003), a dynamic congestion pricing model is formulated as a bi-level program, and the prices are allowed to affect the route and departure time choices of travelers. It is assumed that the prices are equal across the links that can be charged.

5.1.4 Objectives and Contributions

In this chapter, we develop methods for dynamic congestion pricing in general traffic networks. The main purpose of restudying the problem is to address some of the existing limitations in the literature on dynamic pricing. Our approach follows a link-based second-best pricing strategy by allowing some links to remain unpriced and imposing link-specific upper bounds on the prices. Moreover, the prices are allowed to vary by link. We model the congestion pricing problem as a game between a traffic authority, that

sets prices to optimize some system performance measure, and users that react to the implemented prices by choosing paths that minimize their disutilities. If the traffic authority takes into account the reaction of the users while selecting a control strategy (pricing), this would result in a higher payoff function (e.g. higher net social benefit) than if the user reaction function were not considered. This is known as a Stackelberg game (von Stackelberg (1934)) in game theory, and can be formulated as a bi-level program.

In this chapter, we use this methodology to formulate and solve the dynamic congestion pricing problem where the objective is to minimize total travel time given a subset of links that can be priced and upper bounds on the prices. We study users' reactions to the prices at the levels of both route and departure time choices. The model assumes that travel demand and arc capacities are time-dependent (within a day) but stationary from day to day (i.e. the same time-dependent patterns of travel demands, as well as arc capacities, are experienced daily), and that travelers have learned the optimal travel choices through their daily explorations of the network. The developed methods can then be used in a planning (offline) context.

We extend the methodology developed for congestion pricing to the case of emission pricing, where the prices differ as well by vehicle category. While congestion and emission pricing try to reduce the congestion and emission externalities, respectively, it is not immediately obvious how a congestion pricing policy will affect emissions, and vice versa. Thus, there is need to consider both the congestion and emissions criteria when determining the prices. Therefore, we modify the basic congestion pricing model proposed earlier in this chapter to account for certain environmental constraints, such as total emissions and hot spot constraints.

The main contributions of this research can be summarized as follows: (1) the formulation of a dynamic link-based second-best congestion pricing model as a bi-level program, where the prices are time and link-dependent and are constrained by link-specific upper bounds, (2) the development of solution algorithms for the congestion pricing model with both route and departure time choices, (3) the extension of the developed methodology to study emission pricing, (4) the formulation of a congestion (emission) pricing model with total emissions (total travel time) constraints or hot spot

environmental constraints, and (5) the evaluation of the proposed models and algorithms on small hypothetical network examples.

5.1.5 Chapter Organization

The remainder of this chapter is organized as follows. In Section 5.2, we study a dynamic link-based congestion pricing model. Specifically, in Section 5.2.1, we formulate the model. In Section 5.2.2, we review models of route and departure time choices that will be needed later in the chapter to model users' reaction to the prices. In Section 5.2.3, we study the analytical properties of the problem and develop gradient expressions that will be useful in the development of iterative solution algorithms in Section 5.2.4. In Section 5.3, we extend the basic methodology developed for congestion pricing to study emission pricing. In Section 5.4, we study a constrained version of the congestion/emission pricing model. We provide a taxonomy of constraints that can be added to the model, and formulate two variants: congestion (emissions) pricing subject to a maximum total emissions rate (total travel time), and congestion/emission pricing subject to hot spot constraints. In Section 5.5, we evaluate the effectiveness of the developed methods by conducting experimental analyses on small hypothetical network examples. Finally, in Section 5.6, we give conclusions and directions for future research.

5.2 Dynamic Congestion Pricing Model

5.2.1 Formulation

Before presenting the formulation of the congestion pricing model, we summarize in Table 5.1 the basic variables and parameters used in this chapter.

Table 5.1. Notation.

Symbol	Definition
a	link index
$c_{ma}(t)$	cost of link a at time t as perceived by class m users
$c_m^{rs}(p,t)$	cost perceived by class m users departing path p connecting O-D pair (r,s) at time t
$CF^{rs}(p)$	commonality factor for path p of O-D pair (r,s)

$\text{corr}(U(p,t),U(p',t))$	correlation of the total utilities of alternatives (p,t) and (p',t) that share a common departure time
$d^{rs}(p,t)$	travel time of path p connecting O-D pair (r,s) at time departure time t
D_m^{rs}	total demand for travel between O-D pair (r,s) for class m users over the possible set of departure times T
$D_m^{rs}(t)$	demand for travel between O-D pair (r,s) at time t for class m users
$e_m^{rs}(p,t)$	total emissions (grams) per vehicle of category m departing path p connecting O-D pair (r,s) at time t
$EF_{ma}(t)$	emission factor (grams/second) due to vehicle category m entering link a at time t
$E(q)$	total network emissions
$h_m^{rs}(p,t)$	the flow rate of class m users departing path p connecting O-D pair (r,s) at time t
$h_m^{rs*}(p,t)$	the equilibrium flow rate of class m users departing path p connecting O-D pair (r,s) at time t
k	time index
l	time index
L_p	length of path p
L_{yp}	length of links common to paths y and p
m	user class index
p	path index
$P_m^{rs}(p)$	marginal probability for class m users of choosing path p connecting O-D pair (r,s) (summed over all departure times)
$P_m^{rs}(p,t)$	joint probability for class m users of choosing path p, connecting O-D pair (r,s), *and* departure time t (for joint choice of route and departure time); probability for class m users of choosing path p, connecting O-D pair (r,s), *at* departure time t (for route choice only)
$P_m^{rs}(p/t)$	probability for class m users of choosing path p connecting O-D pair (r,s) conditional on choosing departure time t
$P_m^{rs}(t)$	marginal probability for class m users of choosing departure time t to travel from origin r to destination s (summed over all paths connecting (r,s))
$P_m^{rs}(t/p)$	probability for class m users of choosing departure time t to travel from origin

	r to destination s conditional on choosing path p
$q_a(t)$	toll levied at the entrance of link a at time t (for congestion pricing)
$q_a^{\max}$	maximum toll that can be levied on link a at any time (for congestion pricing)
$q_{ma}(t)$	toll levied at the entrance of link a at time t for vehicle category m (for emission pricing)
$q_{ma}^{\max}$	maximum toll that can be levied on link a at any time for vehicle category m (for congestion pricing)
$q^{rs}(p,t)$	total toll for users departing path p connecting O-D pair (r,s) at time t (for congestion pricing)
$q_m^{rs}(p,t)$	total toll for vehicle category m departing path p connecting O-D pair (r,s) at time t (for emission pricing)
(r,s)	O-D pair index
R^{rs}	set of routes (paths) between O-D pair (r,s)
t	time index
t^{rs*}	preferred arrival time at destination s for travel between O-D pair (r,s)
$\left[t^{rs*}-\Delta^{rs}, t^{rs*}+\Delta^{rs}\right]$	preferred arrival time window at destination s for travel between O-D pair (r,s)
T	set of departure times
$U_m^{rs}(p,t)$	total utility of (path, departure time) alternative (p,t) for class m users traveling between O-D pair (r,s)
$V_m^{rs}(p,t)$	total systematic component of utility of (path, departure time) alternative (p,t) for class m users traveling between O-D pair (r,s)
$\tilde{V}_m^{rs}(p)$	systematic component of utility common to all (path, departure time) alternatives using path p for class m users traveling between O-D pair (r,s)
$\tilde{V}_m^{rs}(p,t)$	systematic component of utility specific to (path, departure time) alternative (p,t) for class m users traveling between O-D pair (r,s)
$\tilde{V}_m^{rs}(t)$	systematic component of utility common to all (path, departure time) alternatives using departure time t for class m users traveling between O-D pair (r,s)
y	path index
$Z(q)$	total network travel time
α	logit scale parameter
α_{md}	disutility coefficient of a unit of travel time for user class m
α_{mq}	disutility coefficient of a unit of toll for user class m

β_0	parameter used in the C-logit route choice model
γ	parameter used in the C-logit route choice model
Δ^{rs}	indifference band for travel between O-D pair (r,s)
$\varepsilon_m^{rs}(p,t)$	total unobserved component of utility of (path, departure time) alternative (p,t) for class m users traveling between O-D pair (r,s)
$\tilde{\varepsilon}_m^{rs}(p)$	unobserved component of utility common to all (path, departure time) alternatives using path p for class m users traveling between O-D pair (r,s)
$\tilde{\varepsilon}_m^{rs}(p,t)$	unobserved component of utility specific to (path, departure time) alternative (p,t) for class m users traveling between O-D pair (r,s)
$\tilde{\varepsilon}_m^{rs}(t)$	unobserved component of utility common to all (path, departure time) alternatives using departure time t for class m users traveling between O-D pair (r,s)
θ_m	value of time of user class m
μ_p	scale parameter of the utilities in the nested logit model at the path choice level
μ_t	scale parameter of the utilities in the nested logit model at the departure time choice level
τ_{ap}^{rs}/t	arrival time at the entrance of link a if one departs path p connecting O-D pair (r,s) at time t

As discussed in Section 5.1, congestion pricing aims at maximizing net social benefit or minimizing total travel time. In this section, we focus on the latter objective. Before presenting the formulation of the problem, we make two assumptions. First, O-D travel demand is inelastic (to the prices). We study two separate cases: (i) O-D demand is known as a function of time. Commuters can then react to the prices by adjusting their route choice only. (ii) O-D demand is unknown *a priori* as a function of time. In this context, a given fixed number of users demand travel between every O-D pair over a period of time. Commuters can react to the prices by adjusting both their route and departure time choices. The allocation of demand to the departure times in the period of interest is captured in a departure time choice model. In the second assumption, users are unguided during their trips. Thus, the users' reaction function (user equilibrium) can be modeled using an offline dynamic traffic assignment (DTA) model. The problem can

then be formulated as the following bi-level program, which will be referred to as program (A):

$$\text{Min } Z(q) = \sum_t \sum_m \sum_{r,s} \sum_p d^{rs}(p,t) * h_m^{rs*}(p,t) \tag{5.1}$$

Subject to:

$$0 \le q_a(t) \le q_a^{\max} \quad \forall (a,t); \tag{5.2}$$

$$h_m^{rs*}(p,t) \text{ is a solution to a DTA model.} \tag{5.3}$$

The upper level of this bi-level program consists of minimizing the total travel time $Z(q)$, which is the sum over all departure times t, user classes m, O-D pairs (r,s), and paths p. The toll $q_a(t)$ charged at the entrance of link a at time t must be less than or equal to a given upper bound $q_a^{\max}$. Note that $q_a^{\max}$ could also be made time-dependent. The path flows are the solutions of a DTA model, which is the lower level program. This latter model captures the users' reaction (user equilibrium) to tolls. In principle, any DTA model with the following functionalities can be used to find user equilibrium: (1) the ability to represent multiple user classes; this is especially important when users have varying values of time which imply different sensitivities (in terms of route and departure time choices) to the levied tolls, and (2) the route and departure time choice models used in the DTA should possess a general utility function, as the perceived cost of traveling should account for the levied tolls (among other costs such as travel times and/or schedule delays).

Before presenting the analytical properties and solution algorithms for the congestion pricing model, we review the literature and the behavioral assumptions underlying the route and departure time choice problems as these will be needed in the analysis later in the chapter. The developments in this chapter as well as most of the reviewed models assume a stochastic user behavior (in terms of route and departure time choices) since it provides a more realistic representation of user behavior than its deterministic counterpart, and it simplifies the derivations of the analytical expressions later in this chapter.

5.2.2 Route and Departure Time Choice Models

The random utility approach is one popular type of decision rules for modeling consumer behavior (Ben-Akiva and Lerman (1985)). Under this approach, a utility or attractiveness measure is associated with every alternative in the choice set of an individual, and the probability of choosing an alternative is equal to the probability that the alternative has the largest utility. In the most general case, the choice set of an individual traveling between an O-D pair (r,s) is multi-dimensional, and consists of all combinations of routes, departure times, and modes that are available or known to the individual. In this research, mode choice is not considered. To evaluate the attractiveness of the various alternatives for travel between an O-D pair (r,s), one then needs an expression for the total utility of every path $p \in R^{rs}$, where R^{rs} is the set of routes connecting O-D pair (r,s), at every departure time $t \in T$, where T is the set of departure times, and for every user class m. We first review modeling practices for route choice only. Then we discuss models of departure time choice as well as of the joint choice of route and departure time.

5.2.2.1 Route Choice Only

If users are assumed to react to the levied tolls by adjusting their route choice but not departure time choice, the choice set of an individual traveler has a single dimension, namely the choice of a path p from a subset of paths R^{rs} connecting O-D pair (r,s) at a certain departure time t. For simplicity, we assume that we can aggregate users with similar characteristics into homogeneous groups and apply the same utility specification within a group, which is referred to as a class. The utility of a path p at departure time t for user class m is denoted as $U_m^{rs}(p,t)$. Its expression is:

$$U_m^{rs}(p,t) = V_m^{rs}(p,t) + \varepsilon_m^{rs}(p,t), \tag{5.4}$$

where $V_m^{rs}(p,t)$ is the systematic component of utility and $\varepsilon_m^{rs}(p,t)$ is a random error term that represents unobserved effects.

In the numerical results reported later in this chapter, we adopt a simple specification of the utility of a path p at departure time t that includes the path travel time and toll as follows:

$$U_m^{rs}(p,t) = \alpha_1 \left(d^{rs}(p,t) + \frac{1}{\theta_m} q^{rs}(p,t) \right) + \varepsilon_m^{rs}(p,t), \tag{5.5}$$

where θ_m is the value of time of user class m and $q^{rs}(p,t)$ is the total toll levied if one departs path p at time t. It is given by:

$$q^{rs}(p,t) = \sum_{a \in p} q_a\left(\tau_{ap}^{rs} / t\right), \tag{5.6}$$

where τ_{ap}^{rs} / t is the arrival time at the entrance of link a if one departs path p at time t. τ_{ap}^{rs} / t is given by the following recursive relationship:

$$\begin{cases} \tau_{ap}^{rs} / t = t & \text{if } a \text{ is the first link on path } p \\ \tau_{ap}^{rs} / t = \tau_{a'p}^{rs} / t + d_{a'}\left(\tau_{a'p}^{rs} / t\right) & \text{if } a \text{ is after link } a' \text{ on path } p. \end{cases} \tag{5.7}$$

Examples of stochastic models of route choice based on random utility theory are logit, probit, C-logit, and PS-logit (Ben-Akiva and Bierlaire (1999)). The C-logit model, proposed by Cascetta et al. (1996), is a modified multinomial logit (MNL) model that retains the computational properties of MNL and at the same time overcomes its main shortcoming of predicting unrealistic probabilities for highly overlapping paths. This is done by adding a commonality factor to the systematic component of utility. One specification of the commonality factor is:

$$CF^{rs}(p) = \beta_0 \ln \sum_{y \in R^{rs}} \left(\frac{L_{yp}}{\sqrt{L_y L_p}} \right)^{\gamma}, \tag{5.8}$$

where L_p is the length (measured in units of distance) of path p, L_{yp} is the length of links common to paths y and p, and β_0 and γ are parameters that need to be calibrated. The probability of choosing a path p at departure time t is then given by:

$$P_m^{rs}(p,t) = \frac{\exp\left(V_m^{rs}(p,t) - CF^{rs}(p)\right)}{\sum_{y \in R^{rs}} \exp\left(V_m^{rs}(y,t) - CF^{rs}(y)\right)}. \tag{5.9}$$

5.2.2.2 Route and Departure Time Choice

Modeling the departure time choice of users is important in the context of congestion pricing as there is evidence that most of the anticipated benefits of pricing are the result

of departure time shifts (Arnott et al. (1990), van Vuren et al. (1998)). When users are assumed to react to the levied tolls by adjusting both their route and departure time choices, the choice set of an individual traveling between O-D pair (r,s) is multi-dimensional and consists of all possible combinations $(p,t) \in R^{rs} \times T$ of paths and departure times. Note that in this chapter we study the *choice* of departure time as a reaction to the levied tolls rather than the *change* in departure time, which arises for instance in the context of traveler information systems (Ben-Akiva and Bierlaire (1999)) and is more relevant in a real-time context as compared to an offline application. In other words, we do not model deviations from an assumed habitual behavior in departure time decisions but rather allocate a given total O-D demand to the departure time alternatives in the period of study. Moreover, we do not consider "macro" shifts between the peak and off-peak periods but rather focus on the rescheduling of trips within the peak period.

Most departure time choice models found in the literature are based on the concept of schedule delay, defined as the difference between the desired and actual arrival times at the destination. Let $\left[t^{rs*} - \Delta^{rs}, t^{rs*} + \Delta^{rs}\right]$ be a preferred arrival time (PAT) window for travel between O-D pair (r,s), where t^{rs*} is a preferred arrival time and Δ^{rs} represents an indifference band or a measure of work start time flexibility. Arrivals outside this window incur schedule disutilities, and late arrivals are generally considered more onerous than early arrivals. Moreover, there exists a trade-off between scheduling disutility and travel time incurred. That is, a traveler who arrives within his PAT window incurs a larger travel time delay (corresponding to travel during congested peak periods) than one who arrives early or late.

Early efforts in departure time choice modeling include the works of Vickrey (1969), Hendrickson and Kocur (1981), and Mahmassani and Herman (1984) who used a deterministic user equilibrium approach and Cosslett (1977), Abkowitz (1980), Small (1982), de Palma et al. (1983), and Ben-Akiva et al. (1986) who used a stochastic user equilibrium approach. See Alfa (1986) for a review of these models. The basic cost or utility specification in these models considers travel time, early schedule delay, and late schedule delay. Variations to this specification include the probability of being late (Cosslett (1977)), reliability terms (Abkowitz (1980)), socioeconomic factors and dummy

variables for lateness (Small (1982)), and out-of-pocket costs (Ben-Akiva et al. (1986)). Mahmassani and Chang (1985, 1986, 1987) used the boundedly rational principle to model the dynamics of departure time choice. More recently, the UK Department of the Environment, Transport, and the Regions (DETR) has recognized the importance of departure time choice modeling and has been active in commissioning research on this topic (see for example Bates (1996), Hyman (1997), van Vuren et al. (1998), and Hague Consulting Group et al. (2002)).

In this work, to generate a feasible set of departure times, the continuous time is discretized. Next, if a stochastic user equilibrium approach is adopted, a model must be chosen to predict the probabilities of various departure time intervals. The multinomial logit model (MNL) has been a common approach to model departure time choice (e.g. Small (1982), Cascetta et al. (1992)). However, the possible correlation among adjacent departure time intervals casts doubt on the validity of the MNL probabilities. For instance, in Small (1987), an Ordered Generalized Extreme Value model is proposed, which captures the correlation among adjacent periods.

The joint choice of route and departure time has been studied in Mahmassani and Herman (1984) for an idealized situation of two parallel routes, using speed-density relationships to model link travel times and using a deterministic equilibrium model to represent route choice. In Ben-Akiva et al. (1986), the day-to-day evolution of departure time patterns on a network with parallel routes connecting a single O-D pair is modeled through a set of difference equations. The within-day route and departure time choices are given by a nested logit model, with the decision of whether to use the network or not at the upper level, the departure time choice at the middle level, and the route choice at the lower level. In Cascetta et al. (1992), the joint choice of route and departure time is modeled as a function of travel time, safety, and comfort. In Antoniou (1997), a nested logit model is developed to predict deviations from habitual travel choices under the provision of information. Given that a traveler has decided to change both route and departure time, the choice of a certain route and departure time combination is modeled by a joint logit model under the assumption that both choices are made simultaneously. In Ran et al. (1996), a deterministic user optimal route and departure time choice model is formulated using a link-based variational inequality for a general network. A probit

model is used in Liu and Mahmassani (1998) for modeling departure time change where day-to-day correlation is assumed.

In general, different tree structures are possible for modeling the joint choice of route and departure time depending on the behavioral assumptions made and the significance of unobserved effects (Cascetta (2001)). For this purpose, we rewrite the utility expression of a path p at departure time t for user class m as follows:

$$\begin{aligned} U_m^{rs}(p,t) &= V_m^{rs}(p,t) + \varepsilon_m^{rs}(p,t) \\ &= \tilde{V}_m^{rs}(p) + \tilde{V}_m^{rs}(t) + \tilde{V}_m^{rs}(p,t) + \tilde{\varepsilon}_m^{rs}(p) + \tilde{\varepsilon}_m^{rs}(t) + \tilde{\varepsilon}_m^{rs}(p,t), \end{aligned} \tag{5.10}$$

where $\tilde{V}_m^{rs}(p)$ is the systematic component of utility common to all alternatives using path p, $\tilde{V}_m^{rs}(t)$ is the systematic component of utility common to all alternatives using departure time t, $\tilde{V}_m^{rs}(p,t)$ is the systematic component of utility specific to the combination (p,t), and $\tilde{\varepsilon}_m^{rs}(p)$, $\tilde{\varepsilon}_m^{rs}(t)$, $\tilde{\varepsilon}_m^{rs}(p,t)$ are the unobserved components of total utility attributed to path, departure time, and (path, departure time) combinations, respectively.

As an example of these specifications, the systematic component of utility $\tilde{V}_m^{rs}(p)$ specific to a path p can include attributes such as path length, number of signalized intersections, number of left turns on the path, etc. The systematic component of utility $\tilde{V}_m^{rs}(t)$ specific to a departure time t can include variables such as early or late departures in the context of departure time change modeling. The systematic component of utility $\tilde{V}_m^{rs}(p,t)$ specific to the combination (p,t) can include variables such as travel time, toll, and schedule delay (early and late arrivals) when departing path p at time t. In the numerical results reported in this chapter, we adopt the following utility specifications (see Ben-Akiva and Bierlaire (1999) for a discussion of the most relevant attributes in the context of route and departure time choices):

$$\begin{cases} \widetilde{V}_m^{rs}(p) = -CF^{rs}(p) \\ \widetilde{V}_m^{rs}(t) = 0 \\ \widetilde{V}_m^{rs}(p,t) = \alpha_1\left(d^{rs}(p,t) + \dfrac{1}{\theta_m} q^{rs}(p,t)\right) + \alpha_2\left(t^{rs*} - \Delta^{rs} - \left(t + d^{rs}(p,t)\right)\right)^+ \\ \qquad\qquad + \alpha_3\left(t + d^{rs}(p,t) - \left(t^{rs*} + \Delta^{rs}\right)\right)^+, \end{cases} \tag{5.11}$$

where $x^+ = \max\{0, x\}$.

If the shared unobserved effects of alternatives sharing a common path or a common departure time, $\widetilde{\varepsilon}_m^{rs}(p)$ and $\widetilde{\varepsilon}_m^{rs}(t)$, respectively, are negligible, one can use a joint logit model which implies that users choose their routes and departure times simultaneously from a choice set $R^{rs} \times T$ (see Figure 5.1). If however, the variance of either $\widetilde{\varepsilon}_m^{rs}(p)$ or $\widetilde{\varepsilon}_m^{rs}(t)$ is different from zero, then a nested logit model can be used (Ben-Akiva and Lerman (1985)). The case where $\text{var}\left(\widetilde{\varepsilon}_m^{rs}(p)\right) = 0$ corresponds to the tree structure shown in Figure 5.2, which implies that users choose their departure times first and then choose a path conditional on departure time. Alternatively, $\text{var}\left(\widetilde{\varepsilon}_m^{rs}(t)\right) = 0$ corresponds to the tree structure shown in Figure 5.3, which implies that users choose their paths first and then their departure times conditional on the chosen path.

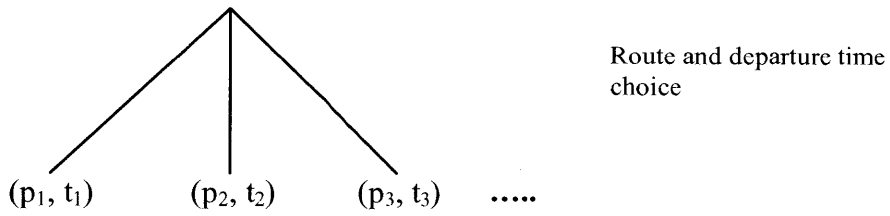


Figure 5.1. Joint logit model of route and departure time choice.

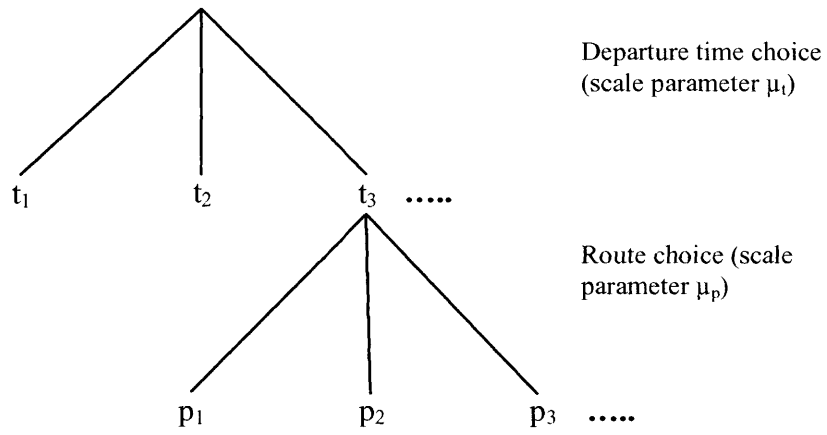


Figure 5.2. Nested logit model 1 of route and departure time choice with departure time choice at the upper level and route choice at the lower level.

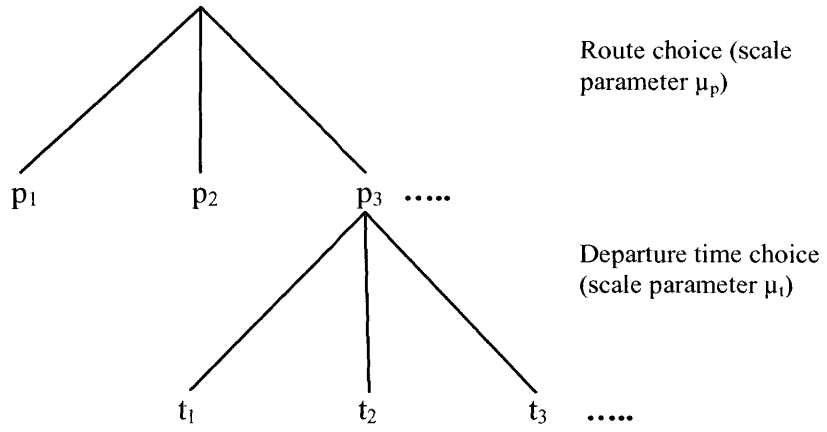


Figure 5.3. Nested logit model 2 of route and departure time choice with route choice at the upper level and departure time choice at the lower level.

Table 5.2 shows the joint, marginal, and conditional probabilities corresponding to the joint logit (Table 5.2 a) and the two nested logit models (Table 5.2 b) depicted above. For simplicity, the indices referring to user class and O-D pair are dropped in Table 5.2. In the probability expressions given in Table 5.2, the terms $V'(p)$ and $V'(t)$ have been referred to as measures of inclusive value (McFadden (1978)) or accessibility (Ben-Akiva and Lerman (1979)), and they represent the expected maximum utility of alternatives in nests p and t, respectively. Their expressions are:

$$\begin{cases} \begin{cases} V'(p) = \ln \sum_t \exp\left(\tilde{V}(t) + \tilde{V}(p,t)\right) \\ V'(t) = \ln \sum_p \exp\left(\tilde{V}(p) + \tilde{V}(p,t)\right) \end{cases} & \text{for joint logit model} \\ V'(t) = \frac{1}{\mu_p} \ln \sum_p \exp\left(\mu_p \left(\tilde{V}(p) + \tilde{V}(p,t)\right)\right) & \text{for nested logit model 1} \\ V'(p) = \frac{1}{\mu_t} \ln \sum_t \exp\left(\mu_t \left(\tilde{V}(t) + \tilde{V}(p,t)\right)\right) & \text{for nested logit model 2.} \end{cases} \tag{5.12}$$

The parameters μ_p and μ_t are scale parameters of the utilities associated with path and departure time alternatives, respectively. Only the ratio $\frac{\mu_p}{\mu_t}$ (or $\frac{\mu_t}{\mu_p}$) is identifiable, and so it is common practice to normalize either μ_p or μ_t to one. Moreover, for nested logit model 1, the scale parameters should satisfy $0 \le \frac{\mu_t}{\mu_p} \le 1$, since $\frac{\mu_t}{\mu_p}$ can be shown to be equal to $\sqrt{1 - \text{corr}(U(p,t), U(p',t))}$ (Ben-Akiva and Lerman (1985)) (and hence less than or equal to one), where $\text{corr}(U(p,t), U(p',t))$ is the correlation of the total utilities of alternatives (p,t) and (p',t) that share a common departure time. Similarly, for nested logit model 2, $0 \le \frac{\mu_p}{\mu_t} \le 1$.

Table 5.2. Probability expressions predicted by the joint logit (part a) and nested logit models (part b) for the choice of route and departure time.

(a)

Probability	Joint Logit
$P(p,t)$	$\frac{\exp\left(\tilde{V}(p) + \tilde{V}(t) + \tilde{V}(p,t)\right)}{\sum_{(p',t')} \exp\left(\tilde{V}(p') + \tilde{V}(t') + \tilde{V}(p',t')\right)}$
$P(p)$	$\frac{\exp\left(\tilde{V}(p) + V'(p)\right)}{\sum_{p'} \exp\left(\tilde{V}(p') + V'(p')\right)}$
$P(t)$	$\frac{\exp\left(\tilde{V}(t) + V'(t)\right)}{\sum_{t'} \exp\left(\tilde{V}(t') + V'(t')\right)}$

$P(p/t)$	$\dfrac{\exp\left(\tilde{V}(p)+\tilde{V}(p,t)\right)}{\sum_{p'}\exp\left(\tilde{V}(p')+\tilde{V}(p',t)\right)}$
$P(t/p)$	$\dfrac{\exp\left(\tilde{V}(t)+\tilde{V}(p,t)\right)}{\sum_{t'}\exp\left(\tilde{V}(t')+\tilde{V}(p,t')\right)}$

(b)

Probability	Nested Logit 1 (Upper level: departure time choice)	Nested Logit 2: (Upper level: route choice)
$P(p,t)$	$\dfrac{\exp\left(\begin{matrix}\mu_p\left(\tilde{V}(p,t)+\tilde{V}(p)\right)+\mu_t\tilde{V}(t)\\+(\mu_t-\mu_p)V'(t)\end{matrix}\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+V'(t')\right)\right)}$	$\dfrac{\exp\left(\begin{matrix}\mu_t\left(\tilde{V}(p,t)+\tilde{V}(t)\right)+\mu_p\tilde{V}(p)\\+(\mu_p-\mu_t)V'(p)\end{matrix}\right)}{\sum_{p'}\exp\left(\mu_p\left(\tilde{V}(p')+V'(p')\right)\right)}$
$P(p)$	$\dfrac{\sum_{t}\exp\left(\begin{matrix}\mu_p\left(\tilde{V}(p,t)+\tilde{V}(p)\right)+\mu_t\tilde{V}(t)\\+(\mu_t-\mu_p)V'(t)\end{matrix}\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+V'(t')\right)\right)}$	$\dfrac{\exp\left(\mu_p\left(\tilde{V}(p)+V'(p)\right)\right)}{\sum_{p'}\exp\left(\mu_p\left(\tilde{V}(p')+V'(p')\right)\right)}$
$P(t)$	$\dfrac{\exp\left(\mu_t\left(\tilde{V}(t)+V'(t)\right)\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+V'(t')\right)\right)}$	$\dfrac{\sum_{p}\exp\left(\begin{matrix}\mu_t\left(\tilde{V}(p,t)+\tilde{V}(t)\right)+\mu_p\tilde{V}(p)\\+(\mu_p-\mu_t)V'(p)\end{matrix}\right)}{\sum_{p'}\exp\left(\mu_p\left(\tilde{V}(p')+V'(p')\right)\right)}$
$P(p/t)$	$\dfrac{\exp\left(\mu_p\left(\tilde{V}(p)+\tilde{V}(p,t)\right)\right)}{\sum_{p'}\exp\left(\mu_p\left(\tilde{V}(p')+\tilde{V}(p',t)\right)\right)}$	$\dfrac{P(p,t)}{P(t)}$
$P(t/p)$	$\dfrac{P(p,t)}{P(p)}$	$\dfrac{\exp\left(\mu_t\left(\tilde{V}(t)+\tilde{V}(p,t)\right)\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+\tilde{V}(p,t')\right)\right)}$

The probability expressions corresponding to the joint logit model (Table 5.2 a) are derived in Ben-Akiva and Lerman (1985) in a context of mode and destination choice. For the nested logit models, the expressions of the marginal probabilities of choices at the upper level ($P(t)$ for nested logit model 1 and $P(p)$ for nested logit model 2) and the expressions of the conditional probabilities of choices at the lower level ($P(p/t)$ for nested logit model 1 and $P(t/p)$ for nested logit model 2) are also derived in Ben-Akiva and Lerman (1985) in a context of mode and destination choice. The joint probabilities $P(p,t)$ in Table 5.2 b are obtained using the following expressions:

$$P(p,t)=\begin{cases}P(t)*P(p/t) & \text{for nested logit model 1}\\ P(p)*P(t/p) & \text{for nested logit model 2.}\end{cases} \tag{5.13}$$

The marginal probabilities $P(p)$ in nested logit model 1 and $P(t)$ in nested logit model 2 are obtained using the following expressions:

$$P(p)=\sum_{t}P(p,t), \tag{5.14}$$

and

$$P(t)=\sum_{p}P(p,t). \tag{5.15}$$

5.2.3 Analytical Properties

Solution algorithms for bi-level optimization problems where traffic equilibrium arises as a subproblem can be developed by conducting a sensitivity analysis (Yang (1997)). Let $q^1=\{q_a^1(l)\}$ be an initial feasible vector of prices (i.e. satisfying the lower and upper bounds on the link prices). Program (A) can then be approximated by the following program (B):

$$\underset{q}{\text{Min}}\, Z(q)=Z(q^1)+(q-q^1)^T\left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^1} \tag{5.16}$$

Subject to:

$$0\le q_a(t)\le q_a^{\max}\quad \forall(a,t). \tag{5.17}$$

In program (B), the objective function is a first-order Taylor series approximation of total travel time, at $q=q^1$. $Z(q^1)$ represents the total travel time corresponding to price vector q^1, and can be computed by solving for a user equilibrium given q^1. $\left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^1}$ is the gradient of the total travel time function with respect to the price vector, evaluated at q^1. Since $Z(q^1)$ and $\left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^1}$ can be evaluated by solving the dynamic traffic assignment problem using the price vector q^1, program (B) is then equivalent to solving the following linear program (C):

$$\underset{q}{\text{Min}}\, Z(q) = q^T \left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^1} \tag{5.18}$$

Subject to:

$$0 \le q_a(t) \le q_a^{\max} \quad \forall (a,t). \tag{5.19}$$

The gradient expression $\frac{\partial Z(q)}{\partial q_a(l)}$ can thus be utilized to predict changes in the total travel time $Z(q)$ when the prices $q_a(l)$ are altered slightly. In this section, we first develop the gradient expression for the case where users are assumed to react to the prices by adjusting their route choice only. Then we extend the analysis to cases where users adjust both their route and departure time choices. Note that it is implicitly assumed that the gradient $\frac{\partial Z(q)}{\partial q_a(l)}$ exists. The validity of this assumption is a question for future research.

5.2.3.1 Gradient Expression for Route Choice Only

Since $Z(q) = \sum_t \sum_m \sum_{r,s} \sum_p d^{rs}(p,t) * h_m^{rs*}(p,t)$, we have:

$$\frac{\partial Z(q)}{\partial q_a(l)} = \sum_t \sum_m \sum_{r,s} \sum_p \left[d^{rs}(p,t) \frac{\partial h_m^{rs*}(p,t)}{\partial q_a(l)} + h_m^{rs*}(p,t) \frac{\partial d^{rs}(p,t)}{\partial q_a(l)} \right]. \tag{5.20}$$

First we find an expression for the term $\frac{\partial h_m^{rs*}(p,t)}{\partial q_a(l)}$. The flow on path p (connecting O-D pair (r,s)) at time t is a function of the utilities of all paths between O-D pair (r,s) at time t. Therefore, we have:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial q_a(l)} = \sum_{y \in R^{rs}} \frac{\partial h_m^{rs*}(p,t)}{\partial V_m^{rs}(y,t)} * \frac{\partial V_m^{rs}(y,t)}{\partial q_a(l)}. \tag{5.21}$$

The term $\frac{\partial V_m^{rs}(y,t)}{\partial q_a(l)}$ is such that:

$$\frac{\partial V_m^{rs}(y,t)}{\partial q_a(l)} = \frac{\partial V_m^{rs}(y,t)}{\partial q^{rs}(y,t)} * \frac{\partial q^{rs}(y,t)}{\partial q_a(l)}$$

$$= \begin{cases} \dfrac{\partial V_m^{rs}(y,t)}{\partial q^{rs}(y,t)} * 1 = \alpha_{mq} & \text{if } a \in y \text{ and } t + d_y^{ra}(t) = l \\ 0 & \text{otherwise,} \end{cases} \tag{5.22}$$

where α_{mq} is the disutility of a unit toll as perceived by class m users and $d_y^{ra}(t)$ is the time needed to travel from origin r to link a if one departs path y at time t. Since $D_m^{rs}(t)$, the demand of class m users for travel between O-D pair (r,s) at time t, is fixed, the term $\dfrac{\partial h_m^{rs*}(p,t)}{\partial V_m^{rs}(y,t)}$ is given by:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial V_m^{rs}(y,t)} = \frac{\partial \left(D_m^{rs}(t) * P_m^{rs}(p,t)\right)}{\partial V_m^{rs}(y,t)} = D_m^{rs}(t) * \frac{\partial P_m^{rs}(p,t)}{\partial V_m^{rs}(y,t)}. \tag{5.23}$$

In the above expression, it is assumed that the probability $P_m^{rs}(p,t)$ of choosing path p at time t for class m users is equal to the proportion of class m users choosing path p at time t, which is a reasonable assumption if the demand $D_m^{rs}(p,t)$ is sufficiently large.

Substituting expressions (5.22) and (5.23) in expression (5.21), we obtain:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial q_a(l)} = \alpha_{mq} D_m^{rs}(t) \sum_{y \in R^{rs} \text{ and } t \in \{z : z \geq 0 \text{ and } z + d_y^{ra}(z) = l\}} \frac{\partial P_m^{rs}(p,t)}{\partial V_m^{rs}(y,t)}. \tag{5.24}$$

For users with stochastic dynamic user-optimal behavior, and for a logit or C-logit route choice model, the term $\dfrac{\partial P_m^{rs}(p,t)}{\partial V_m^{rs}(y,t)}$ is given by:

$$\frac{\partial P_m^{rs}(p,t)}{\partial V_m^{rs}(y,t)} = \begin{cases} P_m^{rs}(p,t)\left[1 - P_m^{rs}(p,t)\right], & \text{if } p = y \\ -P_m^{rs}(p,t) P_m^{rs}(y,t), & \text{if } p \neq y. \end{cases} \tag{5.25}$$

Next we consider the term $\dfrac{\partial d^{rs}(p,t)}{\partial q_a(l)}$, which represents the change in travel time on a path if the toll on a certain link changes slightly. At the beginning of the writing of this thesis, we have assumed that the term $h_m^{rs*}(p,t)\dfrac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ is approximately zero, and we have based the numerical results in Section 5.5 (except for the first example) on this

assumption. Towards the end of the writing of this thesis, we have tried to develop an expression for the term $\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$, which we conjecture to be correct, but whose validity is subject to further verification. Even though the term $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ has been neglected in some of the experiments conducted, the results reported in Section 5.5 indicate that the proposed pricing methods can still achieve savings in total travel time as compared to the no-toll situation.

Below we present the expression that we have derived for the term $\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$. Since the travel time of a path p (connecting O-D pair (r,s)) at time t is a function of all path flows between all O-D pairs and at all times, we can express $\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ as follows:

$$\frac{\partial d^{rs}(p,t)}{\partial q_a(l)} = \sum_{r',s'}\sum_{t'}\sum_{p'}\left[\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')} * \frac{\partial h^{r's'*}(p',t')}{\partial q_a(l)}\right], \tag{5.26}$$

where $h^{r's'*}(p',t')$ is the total equilibrium flow of path p' between O-D pair (r',s') at departure time t'. $h^{r's'*}(p',t')$ is given by:

$$h^{r's'*}(p',t') = \sum_{m'} h_{m'}^{r's'*}(p',t'). \tag{5.27}$$

Therefore, we have:

$$\frac{\partial h^{r's'*}(p',t')}{\partial q_a(l)} = \frac{\partial\left(\sum_{m'} h_{m'}^{r's'*}(p',t')\right)}{\partial q_a(l)} = \sum_{m'}\frac{\partial h_{m'}^{r's'*}(p',t')}{\partial q_a(l)}. \tag{5.28}$$

Substituting expression (5.24) in expression (5.28), we obtain:

$$\frac{\partial h^{r's'*}(p',t')}{\partial q_a(l)} = \sum_{m'}\alpha_{m'q}D_{m'}^{r's'}(t') \sum_{y\in R^{r's'} \text{ and } t'\in\{z:z\geq 0 \text{ and } z+d_y^{r'a}(z)=l\}} \frac{\partial P_{m'}^{r's'}(p',t')}{\partial V_{m'}^{r's'}(y,t')}. \tag{5.29}$$

The derivation of the term $\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')}$ is provided in Appendix E. Here we only summarize the final result given by the following expression:

$$\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')} = \begin{cases} \dfrac{1}{\sum_{m'} \alpha_{m'd} D^{rs}_{m'}(t) \dfrac{\partial P^{rs}_{m'}(p',t)}{\partial V^{rs}_{m'}(p,t)}}, & \text{if } t'=t, (r',s')=(r,s) \\ \text{undefined}, & \text{otherwise}, \end{cases} \tag{5.30}$$

where $\alpha_{m'd}$ is the disutility of a unit of travel time for user class m'.

Substituting expressions (5.29) and (5.30) in expression (5.26), we obtain:

$$\frac{\partial d^{rs}(p,t)}{\partial q_a(l)} = \sum_{p'} \left[\frac{\partial d^{rs}(p,t)}{\partial h^{rs*}(p',t)} * \frac{\partial h^{rs*}(p',t)}{\partial q_a(l)} \right]$$

$$= \sum_{p'} \left(\begin{array}{l} \dfrac{1}{\sum_{m'} \alpha_{m'd} D^{rs}_{m'}(t) \dfrac{\partial P^{rs}_{m'}(p',t)}{\partial V^{rs}_{m'}(p,t)}} \\ * \sum_{m'} \alpha_{m'q} D^{rs}_{m'}(t) \sum_{y \in R^{rs} \text{ and } t \in \{z: z \geq 0 \text{ and } z + d^{ra}_y(z) = l\}} \dfrac{\partial P^{rs}_{m'}(p',t)}{\partial V^{rs}_{m'}(y,t)} \end{array} \right). \tag{5.31}$$

Therefore, the gradient $\dfrac{\partial Z(q)}{\partial q_a(l)}$ can be computed by substituting expressions (5.24) and (5.31) in (5.20). If the term $h^{rs*}_m(p,t) \dfrac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ is assumed to be approximately zero, one could then approximate the gradient expression given in (5.20) as follows:

$$\frac{\partial Z(q)}{\partial q_a(l)} = \sum_t \sum_m \sum_{r,s} \sum_p \alpha_{mq} d^{rs}(p,t) D^{rs}_m(t) \sum_{y \in R^{rs} \text{ and } t \in \{z: z \geq 0 \text{ and } z + d^{ra}_y(z) = l\}} \frac{\partial P^{rs}_m(p,t)}{\partial V^{rs}_m(y,t)}. \tag{5.32}$$

5.2.3.2 Gradient Expressions for Route and Departure Time Choice

As in Section 5.2.3.1, we have:

$$\frac{\partial Z(q)}{\partial q_a(l)} = \sum_t \sum_m \sum_{r,s} \sum_p \left[d^{rs}(p,t) \frac{\partial h^{rs*}_m(p,t)}{\partial q_a(l)} + h^{rs*}_m(p,t) \frac{\partial d^{rs}(p,t)}{\partial q_a(l)} \right].$$

We have assumed that the term $h^{rs*}_m(p,t) \dfrac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ is approximately zero. If users adjust both their route and departure time choices, the flow on path p at time t is a function of the utilities of all paths between O-D pair (r,s) at all times $k \in T$ since all (path, departure time) combinations (p,t) are competing alternatives. Therefore, we have:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial q_a(l)} = \sum_{y \in R^{rs}} \sum_{k \in T} \frac{\partial h_m^{rs*}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)} * \frac{\partial \tilde{V}_m^{rs}(y,k)}{\partial q_a(l)}. \tag{5.33}$$

The term $\frac{\partial \tilde{V}_m^{rs}(y,k)}{\partial q_a(l)}$ can be determined using expression (5.34):

$$\begin{aligned} \frac{\partial \tilde{V}_m^{rs}(y,k)}{\partial q_a(l)} &= \frac{\partial \tilde{V}_m^{rs}(y,k)}{\partial q^{rs}(y,k)} * \frac{\partial q^{rs}(y,k)}{\partial q_a(l)} \\ &= \begin{cases} \frac{\partial \tilde{V}_m^{rs}(y,k)}{\partial q^{rs}(y,k)} * 1 = \alpha_{mq} & \text{if } a \in y \text{ and } k + d_y^{ra}(k) = l \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{5.34}$$

The term $\frac{\partial h_m^{rs*}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}$ can be obtained using expression (5.35):

$$\frac{\partial h_m^{rs*}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)} = \frac{\partial \left(D_m^{rs} * P_m^{rs}(p,t)\right)}{\partial \tilde{V}_m^{rs}(y,k)} = D_m^{rs} * \frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}. \tag{5.35}$$

Note that the total demand D_m^{rs} of user class m for travel between O-D pair (r,s) is assumed to be fixed.

The mathematical expressions of the joint probabilities $P_m^{rs}(p,t)$ depend on the structure of the model used to represent the joint route and departure time decisions. Consequently, different behavioral model structures result in different gradient expressions. Below we present the results for each model; the derivations are given in Appendix E.

Joint Logit Model

$$\frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)} = \begin{cases} P_m^{rs}(p,t)\left[1 - P_m^{rs}(p,t)\right] & \text{if } (p,t) = (y,k) \\ -P_m^{rs}(p,t) P_m^{rs}(y,k) & \text{if } (p,t) \neq (y,k). \end{cases} \tag{5.36}$$

Nested Logit Model 1

$$\frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)} = \begin{cases} \left[\mu_p + (\mu_t - \mu_p) P_m^{rs}(p/t) - \mu_t P_m^{rs}(p,t)\right] P_m^{rs}(p,t) & \text{if } p = y, t = k \\ \left[\mu_t - \mu_p - \mu_t P_m^{rs}(t)\right] P_m^{rs}(y/t) P_m^{rs}(p,t) & \text{if } p \neq y, t = k \\ -\mu_t P_m^{rs}(p,k) P_m^{rs}(p,t) & \text{if } p = y, t \neq k \\ -\mu_t P_m^{rs}(y,k) P_m^{rs}(p,t) & \text{if } p \neq y, t \neq k. \end{cases} \tag{5.37}$$

Nested Logit Model 2

$$\frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)} = \begin{cases} \left[\mu_t + (\mu_p - \mu_t)P_m^{rs}(t/p) - \mu_p P_m^{rs}(p,t)\right]P_m^{rs}(p,t) & \text{if } p = y, t = k \\ -\mu_p P_m^{rs}(y,t)P_m^{rs}(p,t) & \text{if } p \neq y, t = k \\ \left[\mu_p - \mu_t - \mu_p P_m^{rs}(p)\right]P_m^{rs}(k/p)P_m^{rs}(p,t) & \text{if } p = y, t \neq k \\ -\mu_p P_m^{rs}(y,k)P_m^{rs}(p,t) & \text{if } p \neq y, t \neq k. \end{cases} \quad (5.38)$$

In summary, an approximation of the gradient expression in the presence of route and departure time choices is given by:

$$\frac{\partial Z(q)}{\partial q_a(l)} = \sum_t \sum_m \sum_{r,s} \sum_p \alpha_{mq} d^{rs}(p,t) \sum_{y \in R^{rs}} \sum_{k \in \{z: z \geq 0 \text{ and } z + d_y^{ra}(z) = l\}} D_m^{rs} * \frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}, \quad (5.39)$$

where $\frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}$ is given by expression (5.36) for the joint logit model, expression (5.37) for nested logit model 1, and expression (5.38) for nested logit model 2.

5.2.4 Solution Algorithms

Various algorithms can be developed to solve the above models. We first outline a master iterative algorithm for solving the congestion pricing model. Let q^n denote the price vector obtained in the nth iteration. The algorithm starts with an initial feasible price vector q^1 (satisfying lower and upper bounds on the prices), and in every iteration the algorithm finds a user equilibrium solution and computes the gradients to find new prices that can potentially decrease the objective function $Z(q)$. The algorithm terminates after a prespecified number of iterations. The main steps of the algorithm are described below:

Step 1: Initialization

- N = maximum number of pricing iterations; $q^1 = 0$; $n = 1$.

Step 2: DTA

- Given the prices q^n, find dynamic user equilibrium.

Step 3: Compute gradients

- Compute the gradient $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n} \forall (a,l)$.

If $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n} > 0$, then a slight increase in $q_a^n(l)$ results in an increase in total travel time, so we set $q_a^{new}(l) = 0$. If $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n} < 0$, then a slight increase in $q_a^n(l)$ results in a decrease in total travel time, so we set $q_a^{new}(l) = q_a^{\max}$. Otherwise, if $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n} = 0$, we set $q_a^{new}(l) = \frac{q_a^{\max}}{2}$.

Step 4: Update link tolls

- Update link tolls (through the method of successive averages)

$$\alpha^n = \frac{1}{n+1};$$
$$q_a^{n+1}(l) = q_a^n(l) + \alpha^n \left(q_a^{new}(l) - q_a^n(l)\right).$$

Step 5: Stopping Criterion

- If $n = N$, then stop. Otherwise, $n = n+1$, and go to Step 2.

Note that this algorithm is a heuristic. That is, the computed prices are not guaranteed to be (global as well as local) optimal since the objective function is non-convex. Moreover, the solution obtained by the algorithm might be dependent on the initial vector of prices. From a practical standpoint, the numerical results shown later in the chapter indicate that the application of the algorithm to small network examples resulted in savings in total travel time (total emissions) for congestion (emission) pricing as compared to the no-toll situation.

One can develop different specializations of the master algorithm that differ in the computation of the path flows (Step 2) and their derivatives with respect to the tolls since they correspond to different mathematical programs (Step 3) each corresponding to a formulation based on a different DTA model. For instance, the computation of flows and of gradient expressions is dependent on whether route choice only or both route and

departure time choices are modeled. Below we describe these computations for four such specializations of the master algorithm. The first specialization, denoted as Algorithm 1, solves the congestion pricing model with route choice only. The input to Algorithm 1 consists of a network topology, a given time-dependent O-D matrix, a route choice model, and link travel time functions. The three other specializations, denoted as Algorithms 2, 3, and 4, solve the congestion pricing model with both route and departure time choices. The input to Algorithms 2, 3, and 4 consists of a network topology, a given fixed total number of travelers for every O-D pair, a joint route and departure time choice model, and link travel time functions.

5.2.4.1 Algorithm 1

In Algorithm 1, since only route choice but not departure time choice is modeled, path flows (Step 2) are computed according to $h_m^{rs}(p,t) = D_m^{rs}(t) * P_m^{rs}(p,t)$, where $P_m^{rs}(p,t)$ is computed using a C-logit model (expression (5.9)). The gradient $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n}$ (Step 3) is computed using expression (5.32) (as an approximation). Moreover, we assume that if there is no demand that can reach a certain link a at time t, then it is desirable in practice to set the toll charged at the entrance of link a at time t to zero. Therefore, in the computation of the gradient, if $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n} = 0$ and if there is demand that can potentially reach link a at time t, we set $q_a^{new}(l) = \frac{q_a^{\max}}{2}$, otherwise we set $q_a^{new}(l) = 0$.

5.2.4.2 Algorithm 2

In Algorithm 2, the choice model of the DTA is structured as follows. We start with a given total number of travelers for each O-D pair, and we use the expressions of the joint probabilities of choosing (path, departure time) combinations to assign travelers to these combinations in the DTA model. Path flows (Step 2) are computed according to $h_m^{rs}(p,t) = D_m^{rs} * P_m^{rs}(p,t)$, and $P_m^{rs}(p,t)$ is computed using the expressions given in

Tables 5.2 a and 5.2 b. The gradient $\left.\frac{\partial Z(q)}{\partial q_a(t)}\right|_{q=q^n}$ (Step 3) is computed using expression (5.39) (as an approximation). Moreover, when $\left.\frac{\partial Z(q)}{\partial q_a(t)}\right|_{q=q^n} = 0$, then if link a can be reached at time t, we set $q_a^{new}(t) = \frac{q_a^{\max}}{2}$, otherwise we set $q_a^{new}(t) = 0$ since there is no need to charge links during time periods when those links cannot be reached.

5.2.4.3 Algorithm 3

In Algorithm 3, the choice model of the DTA is structured as follows. We start with a given total number of travelers for each O-D pair. We use the expressions of the marginal departure time probabilities to create an O-D matrix. Given the O-D matrix, for every departure time we use the expressions of the path probabilities conditional on the given departure time to assign users to all paths at all departure times. Let I denote the maximum number of departure time model iterations. The various computations within Step 2 of the algorithm can be summarized as follows:

Step 2 a:

- $i = 1$.
- Given the prices q^n and free-flow network conditions (if $n = 1$) or latest network conditions obtained from the DTA (if $n > 1$), compute the marginal departure time probabilities $P_{m(i)}^{rs}(t) \quad \forall (r,s), m, t \in T$ (corresponding to iteration i).

Step 2 b:

- Compute an O-D matrix $D_{m(i)}^{rs}(t) = D_m^{rs} * P_{m(i)}^{rs}(t)$.
- Given the prices q^n and the O-D matrix $D_{m(i)}^{rs}(t)$, find dynamic user equilibrium (route choice only, where $h_m^{rs}(p,t) = D_{m(i)}^{rs}(t) * P_m^{rs}(p/t)$, and $P_m^{rs}(p/t)$ is computed using the expressions given in Tables 5.2 a and 5.2 b).
- Compute the new marginal departure time probabilities $P_{m(new)}^{rs}(t)$.
- Update the marginal departure time probabilities (through the method of successive averages)

$$\alpha^i = \frac{1}{i+1};$$
$$P^{rs}_{m(i+1)}(t) = P^{rs}_{m(i)}(t) + \alpha^i \left(P^{rs}_{m(new)}(t) - P^{rs}_{m(i)}(t)\right).$$

Step 2 c:

- If $i > I$, then go to Step 3. Otherwise, $i = i+1$, and go to Step 2 b.

As in Algorithm 2, the gradient $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n}$ (Step 3) is computed using expression (5.39) (as an approximation), and if $\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^n} = 0$, then $q_a^{new}(l) = \frac{q_a^{\max}}{2}$ if link a can be reached at time t, otherwise $q_a^{new}(l) = 0$.

5.2.4.4 Algorithm 4

In Algorithm 4, the choice model of the DTA is structured as follows. We start with a given total number of travelers for each O-D pair. We use the expressions of the marginal route probabilities to predict the total flow on every path (summed over all departure times). For every path, we use the expressions of the departure time choice probabilities conditional on the given path to allocate the total path flow to different departure time periods. Let I' denote the maximum number of route choice model iterations. The various computations within Step 2 of the algorithm can be summarized as follows:

Step 2 a:

- $i = 1$.
- Given the prices q^n and free-flow network conditions (if $n = 1$) or latest network conditions obtained from the DTA (if $n > 1$), compute the marginal route probabilities $P^{rs}_{m(i)}(p) \quad \forall (r,s), p \in R^{rs}, m$ (corresponding to iteration i).

Step 2 b:

- Compute total (summed over all departure times) path flows for each O-D pair: $h^{rs}_{m(i)}(p) = D^{rs}_m * P^{rs}_{m(i)}(p)$.

- Given the prices q^n and the total path flows $h^{rs}_{m(i)}(p)$, find dynamic user equilibrium (departure time choice only, where $h^{rs}_m(p,t) = h^{rs}_{m(i)}(p) * P^{rs}_m(t/p)$, and $P^{rs}_m(t/p)$ is computed according to the expressions given in Tables 5.2 a and 5.2 b).
- Compute the new marginal route probabilities $P^{rs}_{m(new)}(p)$.
- Update the marginal route probabilities (through the method of successive averages)

$$\alpha^i = \frac{1}{i+1};$$
$$P^{rs}_{m(i+1)}(p) = P^{rs}_{m(i)}(p) + \alpha^i \left(P^{rs}_{m(new)}(p) - P^{rs}_{m(i)}(p)\right).$$

Step 2 c:

- If $i > I'$, then go to Step 3. Otherwise, $i = i+1$, and go to Step 2 b.

The same observations made in Algorithms 2 and 3 regarding the gradient apply to Algorithm 4 as well. Finally, it is an interesting question to investigate whether each of Algorithms 2, 3, and 4 is applicable to all three models of route and departure time choice that we have described before (the joint logit and two nested logit models). We conjecture that the hierarchy of actual choices made (i.e. path before departure time, or vice versa, or simultaneous) need not coincide with the order in which the probabilities are computed in the algorithms. The essential issue in the algorithms is to achieve convergence to a solution consistent with the behavioral assumptions. Thus, for instance, it might be possible to use Algorithm 2, 3, or 4 to solve the congestion pricing model where route and departure time choices are modeled using a joint logit model.

5.3 Emission Pricing

Mobile source emissions continue to be a major contributor to air quality degradation in the U.S. despite several regulatory efforts to reduce emissions, such as stringent emission standards and inspection and maintenance programs (Harrington et al. (1996)). Market-based policies to reduce emissions are receiving more attention from policy-makers since

they offer more flexibility as compared to rigid regulatory programs. Mobile source pricing is believed to be a promising method for reducing pollutant emissions by means of travel demand management (Nagurney (2000 b)).

Pricing aimed at reducing vehicular emissions can take several forms: fuel taxes, pay-at-the pump charges, VMT fees, and emission fees (EPA (1997)). The effectiveness of these pricing strategies varies as they tend to induce different behavioral responses. For instance, fuel taxes might cause a shift to cleaner vehicles or encourage ridesharing. VMT fees impact emission levels by encouraging drivers to drive less. Emission fees charge drivers based on the amount of pollutants their vehicles emit, which is mainly a function of the vehicle type and operating conditions (speed and acceleration). Drivers would react to those prices by improving their vehicles' emission control technology, shifting to less-polluting vehicles, and/or driving less (Deakin and Harvey (1996), EPA (1998)).

Several studies have suggested emission pricing and/or tried to evaluate the potential benefits that can be accrued from a system of emission fees (Harrington et al. (1995, 1996), Kessler and Schroeer (1993), White (1982), Eskeland and Devarajan (1996), Deakin and Harvey (1998), Victoria Transport Policy Institute (2003)). In static networks, the computation of emission fees has been addressed in Nagurney (2000 b), where an environmental standard is also included to ensure a "sustainable" transportation network. However, there has been no systematic study that addresses the computation of those fees in a time-dependent network. In this section, we address the problem of dynamic emission pricing by extending the methodology developed for congestion pricing. Optimizing the levels of all pollutants simultaneously, although desirable in principle, is difficult due to the varying degrees of correlation between speed and pollutants (Emmerink (1998)), and requires a multi-criteria analysis. One way to account for multiple pollutants in the optimization procedure is to transform the emissions due to every emission species to a monetary equivalent, and then minimize the total costs due to all emission species. For a review of estimates of emission costs, the reader is referred to Delucchi (2000). For simplicity, our model will be concerned with the optimization of one generic emission species. Moreover, in our model the prices vary by vehicle category to account for differences in emissions between low and high emitters. The latter can be

accountable for the majority of emissions although they constitute a relatively small proportion of vehicles (Wenzel and Ross (1996)). The problem can be formulated as the following program (D):

$$\text{Min } E(q) = \sum_t \sum_m \sum_{r,s} \sum_p e_m^{rs}(p,t) * h_m^{rs*}(p,t) \tag{5.40}$$

Subject to:

$$0 \le q_{ma}(t) \le q_{ma}^{\max} \quad \forall (a,t); \tag{5.41}$$

$$h_m^{rs*}(p,t) \text{ is a solution to a DTA model.} \tag{5.42}$$

The upper level of this bi-level program consists of minimizing the total emissions mass (grams) $E(q)$ summed over all departure times t, vehicle categories m, O-D pairs (r,s), and paths p, where $e_m^{rs}(p,t)$ denotes the emissions mass per vehicle of category m on path p at departure time t. We impose an upper bound $q_{ma}^{\max}$ on the link toll $q_{ma}(t)$ that varies by vehicle category at entry time t to the link. The lower level is the DTA model.

The emissions $e_m^{rs}(p,t)$ (in grams per vehicle) on a path p at time t can be computed as follows:

$$e_m^{rs}(p,t) = \sum_{a \in p} EF_{ma}\left(\tau_{ap}^{rs} / t\right) * d_a\left(\tau_{ap}^{rs} / t\right), \tag{5.43}$$

where τ_{ap}^{rs} / t is the arrival time at the entrance of link a if one departs path p at time t and $EF_{ma}\left(\tau_{ap}^{rs} / t\right)$ is the emission factor (in grams/second) for vehicle category m if one enters link a at time τ_{ap}^{rs} / t. τ_{ap}^{rs} / t is given by expression (5.7). Emission factors can be obtained using a dynamic emission model (see Cappiello (2002) for a review) integrated with a dynamic traffic model.

The gradient $\dfrac{\partial E(q)}{\partial q_{m'a}(l)}$ can be expressed as follows:

$$\frac{\partial E(q)}{\partial q_{m'a}(l)} = \sum_t \sum_m \sum_{r,s} \sum_p \left[e_m^{rs}(p,t) \frac{\partial h_m^{rs*}(p,t)}{\partial q_{m'a}(l)} + h_m^{rs*}(p,t) \frac{\partial e_m^{rs}(p,t)}{\partial q_{m'a}(l)} \right]. \tag{5.44}$$

The term $\frac{\partial e_m^{rs}(p,t)}{\partial q_{m'a}(l)}$ represents the change in total emissions on a path if the toll on a certain link increases slightly. The increase in tolls affects the utilities of the paths, and hence may affect the path flows, which in turn may affect the path emissions (due to changes in speed and associated estimated accelerations). Similarly to the assumption that was made in the congestion pricing model, we will approximate $h_m^{rs*}(p,t)\frac{\partial e_m^{rs}(p,t)}{\partial q_{m'a}(l)}$ to be zero. Even though this term is neglected, the experimental results on emission pricing reported in Section 5.5 indicate that the proposed emission pricing methods can still achieve savings in total emissions as compared to the no-toll situation.

Next we find an expression for the term $\frac{\partial h_m^{rs*}(p,t)}{\partial q_{m'a}(l)}$. We consider first the case of route choice only. For given values of path utilities (for all O-D pairs), the flow on path p at time t is a function of the utilities of all paths between O-D pair (r,s) at time t. Therefore, we have:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial q_{m'a}(l)} = \sum_{y \in R^{rs}} \frac{\partial h_m^{rs*}(p,t)}{\partial V_m^{rs}(y,t)} * \frac{\partial V_m^{rs}(y,t)}{\partial q_{m'a}(l)}. \tag{5.45}$$

The term $\frac{\partial V_m^{rs}(y,t)}{\partial q_{m'a}(l)}$ is such that:

$$\begin{aligned}\frac{\partial V_m^{rs}(y,t)}{\partial q_{m'a}(l)} &= \frac{\partial V_m^{rs}(y,t)}{\partial q_{m'}^{rs}(y,t)} * \frac{\partial q_{m'}^{rs}(y,t)}{\partial q_{m'a}(l)} \\ &= \begin{cases} \frac{\partial V_m^{rs}(y,t)}{\partial q_{m'}^{rs}(y,t)} * 1 = \alpha_{mq} & \text{if } m = m', a \in y, \text{ and } t + d_y^{ra}(t) = l \\ 0 & \text{otherwise.} \end{cases}\end{aligned} \tag{5.46}$$

The term $\frac{\partial h_m^{rs*}(p,t)}{\partial V_m^{rs}(y,t)}$ is computed using expression (5.23), which we restate for clarity of presentation:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial V_m^{rs}(y,t)} = \frac{\partial \left(D_m^{rs}(t) * P_m^{rs}(p,t)\right)}{\partial V_m^{rs}(y,t)} = D_m^{rs}(t) * \frac{\partial P_m^{rs}(p,t)}{\partial V_m^{rs}(y,t)}. \tag{5.23}$$

Note that $\frac{\partial h_m^{rs*}(p,t)}{\partial q_{m'a}(l)}$ is equal to zero if $m \neq m'$ (see expression (5.46)). Consequently, substituting (5.23) and (5.46) into (5.45) and substituting (5.45) into (5.44), the gradient expression for the case of route choice only is then approximated by:

$$\frac{\partial E(q)}{\partial q_{m'a}(l)} = \sum_t \sum_{r,s} \sum_p \alpha_{m'q} e_{m'}^{rs}(p,t) D_{m'}^{rs}(t) \sum_{y \in R^{rs} \text{ and } t \in \{z : z \geq 0 \text{ and } z + d_y^{ra}(z) = l\}} \frac{\partial P_{m'}^{rs}(p,t)}{\partial V_{m'}^{rs}(y,t)}, \tag{5.47}$$

where for users with stochastic dynamic user-optimal behavior, and for a logit or C-logit route choice model, the term $\frac{\partial P_{m'}^{rs}(p,t)}{\partial V_{m'}^{rs}(y,t)}$ is computed using expression (5.25).

When both route and departure time choices are modeled, a similar analysis shows that the term $\frac{\partial h_m^{rs*}(p,t)}{\partial q_{m'a}(l)}$ is given by:

$$\frac{\partial h_m^{rs*}(p,t)}{\partial q_{m'a}(l)} = \begin{cases} \alpha_{mq} D_m^{rs} \sum_{y \in R^{rs}} \sum_{k \in \{z : z \geq 0 \text{ and } z + d_y^{ra}(z) = l\}} \frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}, & \text{if } m = m' \\ 0, & \text{otherwise.} \end{cases} \tag{5.48}$$

Finally, the gradient $\frac{\partial E(q)}{\partial q_{m'a}(l)}$ can be approximated by the following expression:

$$\frac{\partial E(q)}{\partial q_{m'a}(l)} = \sum_t \sum_{r,s} \sum_p \alpha_{m'q} D_{m'}^{rs} e_{m'}^{rs}(p,t) \sum_{y \in R^{rs}} \sum_{k \in \{z : z \geq 0 \text{ and } z + d_y^{ra}(z) = l\}} \frac{\partial P_{m'}^{rs}(p,t)}{\partial \tilde{V}_{m'}^{rs}(y,k)}, \tag{5.49}$$

where $\frac{\partial P_{m'}^{rs}(p,t)}{\partial \tilde{V}_{m'}^{rs}(y,k)}$ is given by expression (5.36) for the joint logit model, expression (5.37) for nested logit model 1, and expression (5.38) for nested logit model 2.

The solution algorithms (Algorithms 1 to 4) described for the congestion pricing model can also be adapted to solve the emission pricing model. Finally, we note that the extension of the congestion pricing model and algorithms to the case of emission pricing can also be carried out to the case where both congestion and emission are priced in a manner to minimize a weighted sum of total travel time and emissions. It suffices to use the objective function: $Z(q) = \sum_t \sum_m \sum_{r,s} \sum_p \left(\lambda d_p^{rs}(t) + (1-\lambda) e_m^{rs}(p,t)\right) h_m^{rs*}(p,t)$, where λ is a parameter between zero and one.

5.4 Congestion Pricing with Environmental Constraints

In the previous sections of this chapter, we have developed pricing methods to manage congestion, emissions, or a combination of the two criteria. The objective function was to minimize some criterion, and the only constraints were lower and upper bounds on the prices. In this section, we extend the basic framework to include congestion or emission-related constraints that are desirable from a system or a societal point of view. First, we provide a discussion of some of the scenarios that might arise. Then we formulate two variants.

5.4.1 Taxonomy

In Abou Zeid and Chabini (2002), different scenarios for the combined problems of mobility and emissions have been discussed for routing applications from the perspectives of the users of the transport system, the system operators, and the general public. In this section, we discuss some of the scenarios that are relevant for the pricing problem.

5.4.1.1 Scenario 1: Minimize Total Travel Time (Emissions) Subject to an Upper Bound on Total Emissions (Total Travel Time)

The relationship between total travel times and emissions is non-linear and non-monotone. Thus, it is useful to model a case where the objective function is as before, which is to minimize total travel time (total emissions) with one additional constraint: the total emissions (total travel time) summed over all users and times should not exceed a certain upper bound. This problem has been studied in Nagurney (2000 b) for static networks. A system-optimized solution that meets the emissions constraint (referred to as an environmental quality standard in Nagurney (2000 b)) results in a "sustainable" transportation network (Nagurney (2000 b)). In this chapter, we restudy the problem for time-dependent networks and dynamic pricing. The mathematical representation of these additional constraints can be written as follows:

$$E(q) = \sum_t \sum_m \sum_{r,s} \sum_p e_m^{rs}(p,t) * h_m^{rs^*}(p,t) \leq \overline{E}, \tag{5.50}$$

and

$$Z(q) = \sum_t \sum_m \sum_{r,s} \sum_p d^{rs}(p,t) * h_m^{rs^*}(p,t) \leq \overline{Z}. \tag{5.51}$$

Expression (5.50) is an environmental constraint stating that the total emissions should be less than or equal to an upper bound denoted as $\overline{E}$. Expression (5.51) is a total travel time constraint stating that total travel time should be less than or equal to an upper bound denoted as $\overline{Z}$.

5.4.1.2 Scenario 2: Minimize Total Travel Time (or Total Emissions) Subject to Hot Spot Constraints

While the solution generated to the model described in Section 5.4.1.1 ensures that the total network emissions are below a threshold, the solution does not guarantee that the emissions at a local level (e.g. at a link level) are within tolerable limits. We define hot spots as locations in the network where the emission rate exceeds a certain threshold $\overline{H}$. In Scenario 2, the objective function is to minimize total travel time (or total emissions) such that there are no hot spots at any link and at all times. The mathematical representation of this fact translates into adding the following constraint:

$$\sum_m E_{ma}(t) \leq \overline{H} \quad \forall t, \tag{5.52}$$

where $E_{ma}(t)$ denotes the sum of emissions rate of all vehicles of category m that are present on link a at time t.

5.4.1.3 Scenario 3: Minimize Total Travel Time (or Total Emissions) Subject to Air Quality Standards

Air quality standards are defined differently from hot spot constraints. The former are specified using various temporal aggregations. To ensure that these standards are met, the pollutant concentration on every link, averaged over a period of time, should not exceed the upper limit set by EPA for the pollutant. Assume that the standard for the pollutant considered is defined as the average concentration over a period of time of duration W (W depends on the type of the pollutant). This means that at every time t the ambient

pollutant concentration over every link a, averaged over a period of duration W extending from $t-W$ to t, should not exceed the air quality standard. The duration W varies as a function of pollutant type. For instance, for carbon monoxide (CO) and ozone (O_3) the duration over which the standards are defined is typically less than 24 hours to protect against short-term health effects. Longer durations are designed for other pollutants such as sulfur dioxide (SO_2), nitrogen dioxide (NO_2), lead (Pb), and particulate matter with diameters of 10 micrometers or less (PM_{10}) to protect against chronic health effects. For more details about air quality standards, the reader is referred to the website of the Environmental Protection Agency (EPA)'s Office of Air Quality Planning and Standards (http://www.epa.gov/oar/oaqps/). We assume that an average pollutant concentration measured over a time period W for every link a can be transformed into an equivalent average emission rate on a, denoted as $\overline{E}_a$ and expressed in grams/sec. That is, $\overline{E}_a$ is the average emission rate of all vehicles traveling on link a during the period W. Air quality standards would then be met at every time t by bounding the emission rate $\overline{E}_a$, due to all flows on link a aggregated over all times from $t-W$ to t, to an upper limit κ given a knowledge of the emission factor $EF_{ma}(t)$ of vehicle category m on link a at every time t. The mathematical representation of this constraint can be stated as follows:

$$0 \le \overline{E}_a = \frac{\sum_{y=t-W}^{t} \sum_{m} (f_{ma}(y) * EF_{ma}(y) * d_a(y))}{W} \le \kappa \quad \forall (a,t). \tag{5.53}$$

In expression (5.53), $f_{ma}(t)$ represents the flow into link a at time t of category m vehicles. Note that there are no documented values in the literature for the allowable emission rate κ. However, equivalent concentrations that define air quality standards, expressed in units of parts per million (ppm) by volume or milligrams per cubic meter of air (mg/m^3), can be obtained for instance from the website of the Environmental Protection Agency (EPA)'s Office of Air Quality Planning and Standards (http://www.epa.gov/oar/oaqps/).

5.4.1.4 Scenario 4: Minimize Total Travel Time (or Total Emissions) Subject to Equity of Emissions Distributions

Compliance with air quality standards ensures that public health is protected in all areas, but might result in some areas being relatively more polluted than other areas due to higher traffic volumes. For instance, heavily traveled corridors might invoke public complaints (due to the associated emission and noise levels) resulting in the closure of certain streets. It is, therefore, useful to model a scenario where the traffic flow pattern that minimizes total travel time (or emissions) also results in an equitable temporal and/or spatial distribution of emissions in the network. Equity can be enforced by maintaining the difference in emission levels between any two zones within a certain threshold ρ (Gopalan et al. (1990)). If equal emission shares were desired, this threshold would be set to zero. Assume that the traffic network can be divided into Q mutually exclusive zones. Let $\pi_{a,v}(t)$ represent the damage (e.g. health damage, amount of emissions, etc.) caused to zone v due to the flow that travels on link a at time t. The damage function $\pi_{a,v}(t)$ depends on the traffic volume on link a at time t, the proximity of zone v to link a, and the dispersion rate of the pollutant under consideration. The equity of emissions distribution constraint can then be expressed as follows:

$$\sum_t \sum_a \left(\pi_{a,v}(t) - \pi_{a,w}(t)\right) \le \rho \quad \forall\, v,w = 1,2,...,Q. \tag{5.54}$$

5.4.2 Formulations

5.4.2.1 Formulation of Scenario 1

Extending program (A) to include a constraint on the total emissions generated, we obtain a modified program (A'):

$$\text{Min } Z(q) = \sum_t \sum_m \sum_{r,s} \sum_p d^{rs}(p,t) * h_m^{rs*}(p,t) \tag{5.55}$$

Subject to:

$$0 \le q_a(t) \le q_a^{\max} \quad \forall (a,t); \tag{5.56}$$

$$h_m^{rs*}(p,t) \text{ is a solution to a DTA model;} \tag{5.57}$$

$$E(q)=\sum_{t}\sum_{m}\sum_{r,s}\sum_{p}e_{m}^{rs}(p,t)*h_{m}^{rs*}(p,t)\leq \overline{E}\,. \tag{5.58}$$

To obtain a tractable program, as before we linearize the objective function by using a first-order Taylor series approximation of total travel time, at some initial feasible price vector $q=q^{1}$. Note that obtaining an initial feasible price vector is not trivial in this case, as the resulting path flows should satisfy the total emissions constraint given by expression (5.58). We assume that a feasible solution is known. A method to systematically find a feasible solution, should the feasible domain be non-empty, is a useful topic of research. Extending the linear approximation program (C) to include the total emission constraint, we obtain a modified program (C'):

$$\underset{q}{\text{Min}}\, Z(q)=q^{T}\left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^{1}} \tag{5.59}$$

Subject to:

$$0\leq q_{a}(t)\leq q_{a}^{\max}\quad \forall(a,t); \tag{5.60}$$

$$E(q)\leq \overline{E}\,. \tag{5.61}$$

$E(q)$ can be approximated by a Taylor series expansion in the vicinity of the vector q^{1} as follows: $E(q)=E(q^{1})+(q-q^{1})^{T}\left.\frac{\partial E(q)}{\partial q}\right|_{q=q^{1}}$, where $E(q^{1})$ represents the total emissions generated given the price vector q^{1}, and is computed after solving for a user equilibrium given q^{1}. $\left.\frac{\partial E(q)}{\partial q}\right|_{q=q^{1}}$ is the gradient of the total emissions function with respect to the price vector, evaluated at q^{1}, and can be approximated using expressions (5.47) or (5.49), for route choice or route and departure time choices, respectively. Since $E(q^{1})$ and $\left.\frac{\partial E(q)}{\partial q}\right|_{q=q^{1}}$ can be evaluated by solving the dynamic traffic assignment problem using the price vector q^{1}, program (C') is then equivalent to solving linear program (D'):

$$\underset{q}{\text{Min}}\, Z(q) = q^T \left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^1} \tag{5.62}$$

Subject to:

$$0 \le q_a(t) \le q_a^{\max} \quad \forall (a,t); \tag{5.63}$$

$$q^T \left.\frac{\partial E(q)}{\partial q}\right|_{q=q^1} \le \left(\overline{E} - E(q^1) + q^{1T} \left.\frac{\partial E(q)}{\partial q}\right|_{q=q^1} \right) = \widetilde{E}\,. \tag{5.64}$$

We rewrite program (D') in the following form, which we denote as program (E'):

$$\underset{q}{\text{Max}}\, Z(q) = q^T \left(-\left.\frac{\partial Z(q)}{\partial q}\right|_{q=q^1} \right) \tag{5.65}$$

Subject to:

$$0 \le q_a(t) \le q_a^{\max} \quad \forall (a,t); \tag{5.66}$$

$$q^T \left.\frac{\partial E(q)}{\partial q}\right|_{q=q^1} \le \widetilde{E}\,. \tag{5.67}$$

Program (E') is a linear program that can be solved using any linear programming algorithm. Program (E') can also be interpreted as a knapsack problem with a knapsack of capacity $\widetilde{E}$. The aim is to maximize the value of the objects placed in the knapsack while respecting its capacity constraint. There are $K = |A| \times |T'|$ types of objects corresponding to the total number of decision variables (prices for all links and all network time intervals), where $|A|$ is the number of links and $|T'|$ is the number of network time intervals. The weight of an object of type (a,l) (i.e. link a and time l) is equal to $\left.\frac{\partial E(q)}{\partial q_a(l)}\right|_{q=q^1}$, and its value is equal to $-\left.\frac{\partial Z(q)}{\partial q_a(l)}\right|_{q=q^1}$. Moreover, there is a limit $q_a^{\max}$ on the amount $q_a(l)$ of each type of object (a,l) that can be placed in the knapsack. This knapsack problem is different from the traditional knapsack problem in the sense that an object can have a negative weight and/or a negative value.

5.4.2.2 Formulation of Scenario 2

Extending program (A) to include a hot spot constraint, we obtain a modified program (A''):

$$\text{Min } Z(q) = \sum_{t}\sum_{m}\sum_{r,s}\sum_{p} d^{rs}(p,t) * h_m^{rs*}(p,t) \tag{5.68}$$

Subject to:

$$0 \le q_a(t) \le q_a^{\max} \quad \forall (a,t); \tag{5.69}$$

$$h_m^{rs*}(p,t) \text{ is a solution to a DTA model;} \tag{5.70}$$

$$\sum_{m} E_{ma}(t) \le \overline{H} \quad \forall t. \tag{5.71}$$

We further develop program (A'') for a case of a simple network consisting of two parallel routes (Figure 5.4). For simplicity, we assume that there is one vehicle category.

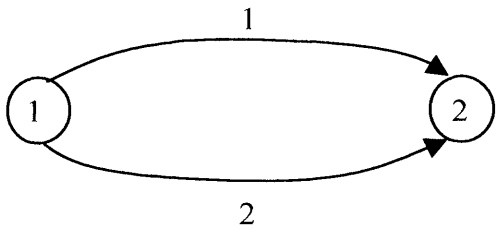


Figure 5.4. Scenario 2 for a two-route example.

The total emission rate on link 1 at time t is given by:

$$E_1(l) = \sum_{t \in \{k: k + d_1(k) \le l \text{ and } k \ge 0\}} [h_1(t) * EF_1(t)] \le \overline{H}, \tag{5.72}$$

where $h_1(t)$ and $EF_1(t)$ are the flow rate and emission factor, respectively, on link 1 at time t. Assuming route choice only, the path flow $h_1(t)$ on link 1 at time t can be expressed as follows:

$$h_1(t) = h_1(q_1, q_2, t) = D_{12}(t) * P_1(t)$$
$$= D_{12}(t) * \frac{\exp\left(-\alpha\left(d_1(t) + \frac{1}{\theta} q_1(t)\right)\right)}{\exp\left(-\alpha\left(d_1(t) + \frac{1}{\theta} q_1(t)\right)\right) + \exp\left(-\alpha\left(d_2(t) + \frac{1}{\theta} q_2(t)\right)\right)}, \tag{5.73}$$

where $D_{12}(t)$ is the travel demand for O-D pair 1-2 at time t, $P_1(t)$ is the probability of choosing route 1 at time t, α is a logit scale parameter, θ is the value of time, and $d_1(t)$ and $q_1(t)$ are the travel time and toll, respectively, on route 1 at time t.

To render the analysis tractable, we use a Taylor series expansion of $h_1(q_1, q_2, t)$ in the vicinity of a feasible price vector q^0, as follows:

$$h_1(q_1, q_2, t) = h_1(q_1^0, q_2^0, t) + (q_1(t) - q_1^0(t)) * \left.\frac{\partial h_1(q_1, q_2, t)}{\partial q_1(t)}\right|_{q^0}$$
$$+ (q_2(t) - q_2^0(t)) * \left.\frac{\partial h_1(q_1, q_2, t)}{\partial q_2(t)}\right|_{q^0}. \tag{5.74}$$

We have:

$$\left.\frac{\partial h_1(q_1, q_2, t)}{\partial q_1(t)}\right|_{q^0} = -\frac{\alpha}{\theta} D_{12}(t) * P_1(q_1^0, q_2^0, t)[1 - P_1(q_1^0, q_2^0, t)], \tag{5.75}$$

and

$$\left.\frac{\partial h_1(q_1, q_2, t)}{\partial q_2(t)}\right|_{q^0} = \frac{\alpha}{\theta} D_{12}(t) * P_1(q_1^0, q_2^0, t) * P_2(q_1^0, q_2^0, t). \tag{5.76}$$

Similarly, we have:

$$\left.\frac{\partial h_2(q_1, q_2, t)}{\partial q_1(t)}\right|_{q^0} = \frac{\alpha}{\theta} D_{12}(t) * P_1(q_1^0, q_2^0, t) * P_2(q_1^0, q_2^0, t), \tag{5.77}$$

and

$$\left.\frac{\partial h_2(q_1, q_2, t)}{\partial q_2(t)}\right|_{q^0} = -\frac{\alpha}{\theta} D_{12}(t) * P_2(q_1^0, q_2^0, t)[1 - P_2(q_1^0, q_2^0, t)]. \tag{5.78}$$

Substituting expressions (5.75) and (5.76) into expression (5.74), we can then express $h_1(t)$ as a linear function of the prices: $h_1(t) = h_1(q_1, q_2, t) = A + Bq_1(t) + Cq_2(t)$, for some constants A, B, and C. Substituting the flow rate $h_1(q_1, q_2, t)$ into expression (5.72), we

would then obtain a linear relationship linking several prices (at different time intervals) and binding them by a capacity constraint, as follows:

$$\sum_{l}\sum_{a=1}^{2}\lambda_a(l)*q_a(l)\le \overline{H}, \tag{5.79}$$

for some constant $\lambda_a(l)$. Since the hot spot constraint should be satisfied for every link and time interval, the number of constraints in the form of equation (5.79) is equal to the number of links times the number of network time intervals. Similarly to program (E') above, the linear approximation congestion pricing model with hot spot constraints can be stated as the following program, denoted as (E''), for a two-link example:

$$\underset{q}{\text{Max}}\, Z(q)=q^T\left(-\frac{\partial Z(q)}{\partial q}\bigg|_{q=q^0}\right) \tag{5.80}$$

Subject to:

$$0\le q\le q^{\max}; \tag{5.81}$$

$$\sum_{l}\sum_{a=1}^{2}\lambda_a(l)*q_a(l)\le \overline{H} \quad \text{for every link and time.} \tag{5.82}$$

Since the hot spot constraint should be satisfied for every link and time, program (E'') can be interpreted as a multiple knapsack problem. As in the previous knapsack problem, the weights and values of objects included in the knapsack can possess negative values.

5.5 Experimental Results

In this section, we report results from several experiments that were conducted on small hypothetical network examples to assess the effectiveness of congestion and emission pricing on total travel times and emissions. We first give a brief overview of the DTA model used to simulate users' reactions to the tolls. Then we present three examples on congestion pricing with route choice only, emission pricing with route choice only, and congestion pricing with both route and departure time choices.

5.5.1 The DTA Model

We adopt the analytical dynamic traffic assignment model described in He (1997) to simulate users' reaction to the implemented prices. This model consists of three main

modules: a user behavioral model, a dynamic network loading model, and a link performance model. Three user classes are modeled: (1) users with fixed route choices, (2) users with stochastic route choices, and (3) users that follow a shortest path. In addition, we have added to the model the option of having different user classes distinguished by different values of time or different vehicle categories. We have also added a departure time choice model. The utility function in the route and departure time choices of users with stochastic behavior is a general utility function that can account for travel time, tolls, and schedule disutility. The dynamic network loading model consists of a set of equations expressing link dynamics, flow propagation, flow conservation, and boundary constraints.

5.5.2 Congestion Pricing with Route Choice Only: a Three-Link Example

5.5.2.1 Network Description

The network used in this example is shown in Figure 5.5. It consists of two O-D pairs: a-c and b-c. O-D pair a-c is connected by one path (link 1), and O-D pair b-c is connected by two paths (path 1: link 2 and path 2: links 3-1). Links 1 and 2 have the same capacity, but link 2 is longer. Thus, several users traveling from b to c use the bridge (link 3) followed by link 1, causing delays to users of O-D pair a-c who have only one travel alternative. In this example, therefore, we investigate the potential savings in travel time from a toll imposed on the bridge (link 3). The link travel time functions are also shown in Figure 5.5 (in minutes, as a function of the link volumes (number of vehicles)).

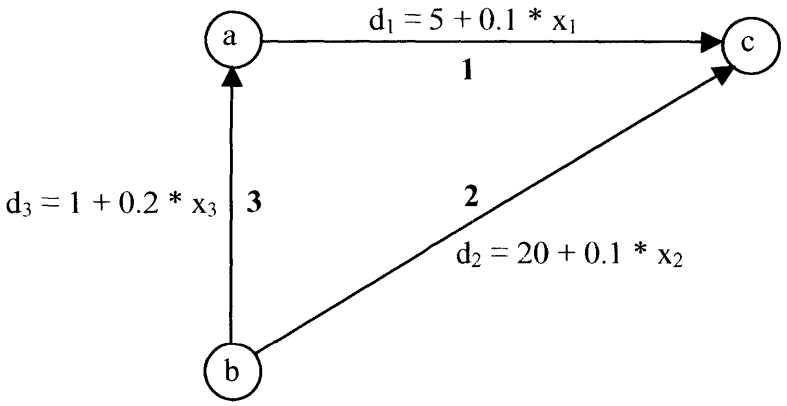


Figure 5.5. Network topology for a three-link example.

5.5.2.2 Scenario Description

The O-D travel demands for O-D pairs a-c and b-c are 1800 veh/hr and 600 veh/hr, respectively. The demand interval is 20 minutes. We let the maximum toll that can be charged on the bridge be equal to $ 3. We assume that all users belong to the stochastic user class. That is, users base their route choices on the perceived rather than the actual travel costs. Within this class, we assume that there are two subclasses of users: one with a high value of time (10 $/hr) and one with a low value of time (3 $/hour). The percentages of users in the two subclasses are 70 % and 30 %, which might be representative of travel during the morning peak period (most trips are home-to-work). In this example, we model users' reactions to the prices in terms of route choice but not departure time choice. The following utility specification is used in the route choice model:

$U_m^{rs}(p,t) = \alpha_1 \left[d^{rs}(p,t) + \frac{1}{\theta_m} q^{rs}(p,t) \right] + \varepsilon_m^{rs}(p,t)$, where $\alpha_1 = -0.106$ min^{-1} (obtained from Small (1982)). The parameters used in the C-Logit model are: $\beta_0 = 1.0$ and $\gamma = 2.0$.

In this example, we also evaluate the impacts of the congestion tolls on the total levels of tailpipe *CO* emissions. The emission factors that we use are shown in Table 5.3. They represent expected emission factors for vehicle category 9 (defined in Cappiello (2002)) and arterials, derived from the integration of EMIT, an instantaneous emission model developed in Cappiello et al. (2002), and a probabilistic acceleration model, developed in Abou Zeid et al. (2002). The emission and acceleration models have been described in Chapter 2 of this thesis.

Table 5.3. Expected tailpipe CO emission factors for vehicle category 9 and arterials. (from Cappiello (2002)).

Speed Range (km/h)	Expected Tailpipe CO Emission Factor (g/s)
0-10	0.0012331
11-20	0.003766
21-30	0.0066965
31-40	0.0095818
41-50	0.0095226
51-60	0.0075375
61-70	0.0087029
71-80	0.011866
81-90	0.0185575
91-100	0.0288626

5.5.2.3 Results

We note that the results shown for this example only correspond to using the full gradient expression (i.e. without neglecting the effect of a slight change in toll on a change in travel time). The results were the same using the exact expression of the gradient or its approximation given by expression (5.32). It appears that the term $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ was not zero for this example (but was rather significant), but that the terms $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ and $d^{rs}(p,t)\frac{\partial h_m^{rs*}(p,t)}{\partial q_a(l)}$ had the same sign, and thus the sign of the gradient (and the subsequent toll setting) was not affected by assuming that the term $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ was zero. The significance of the term $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ should be a subject for future research. Figure 5.6 shows the time-dependent tolls on the bridge (link 3). These tolls cause the total travel time to decrease from 430 veh-hr to 417 veh-hr, which corresponds to a 3 % saving. The total emissions, on the other hand, increase from 6331 grams of CO to 6748 grams (6.6 %). This might be attributed to the improved traffic flow conditions, which lead to higher speeds and higher aggregate emissions (since the expected emission factors in g/s, shown in Table 5.3, are generally a non-decreasing function of speed range). Note that for most of the departure time period the

toll on the bridge approaches its maximum possible value ($ 3). This might suggest that a system-optimal solution can be obtained by diverting most of the users of the bridge (path 2) to path 1 of O-D pair b-c. Setting the toll as large as possible has the effect of decreasing the utility of the bridge as much as possible and hence the probability of using it.

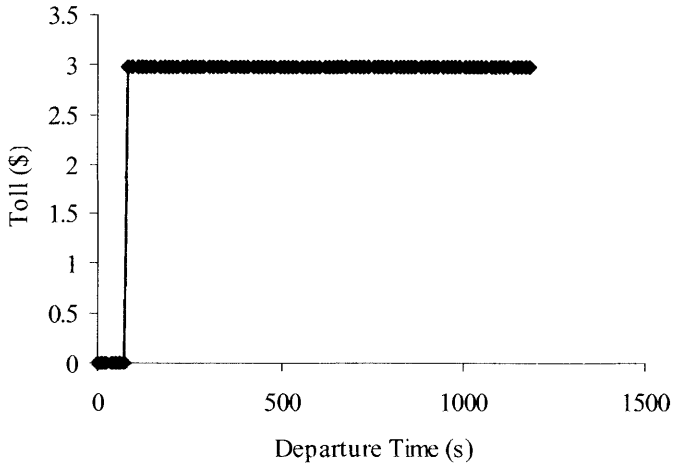
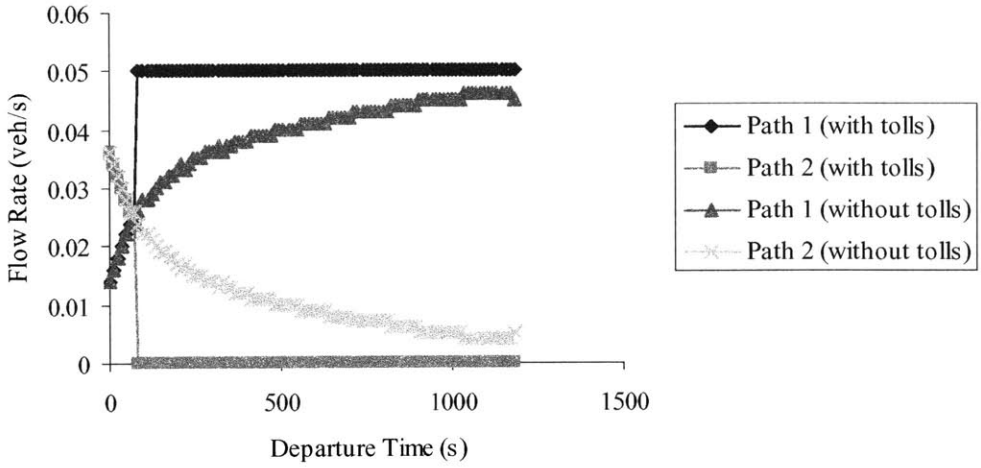
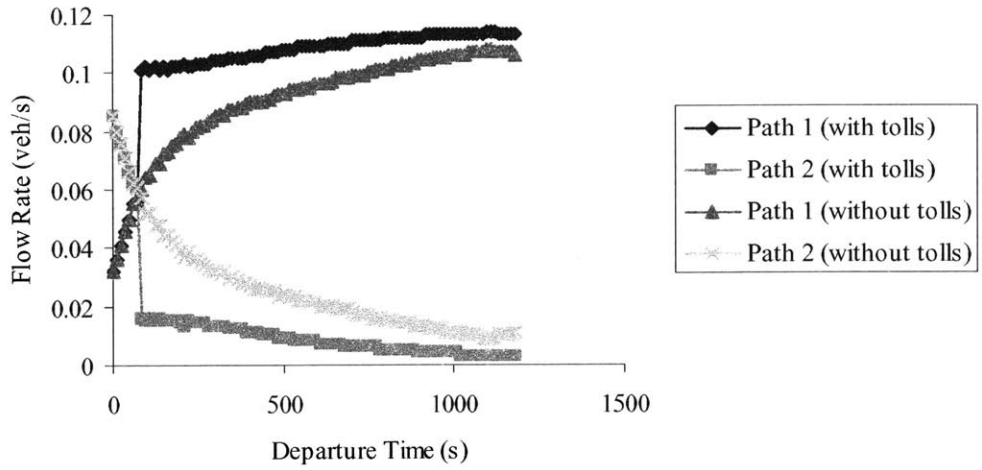


Figure 5.6. Toll on the bridge (link 3).

Next we show the effect of the toll on the path flows. For users of O-D pair a-c, the only available path is link 1, and so the path flow does not vary with the toll on the bridge. For O-D pair b-c, we show the change in path flows separately for users with low value of time ($ 3) and high value of time ($ 10). Figure 5.7 shows the path flows with and without the tolls for users with low (part a) and high (part b) values of time. For both types of users, the flow on path 1 increases and the flow on path 2 decreases after implementing the tolls. However, the sensitivity to the tolls of users with low value of time is higher than that of users with high value of time; the flow rate of users with low value of time drops almost to zero on path 2, while it remains greater than zero for users with high value of time.



(a)



(b)

Figure 5.7. Path flows for O-D pair b-c and users with low value of time (part a) and high value of time (part b).

Finally, Figures 5.8 and 5.9 show the path travel times for O-D pairs a-c and b-c, respectively, with and without the tolls. As expected, the travel time on path 1 of O-D pair a-c decreases in the presence of the tolls. The travel time of path 1 of O-D pair b-c increases in the presence of the tolls due to its larger flow rate, while the travel time of path 2 of O-D pair b-c decreases in the presence of the tolls.

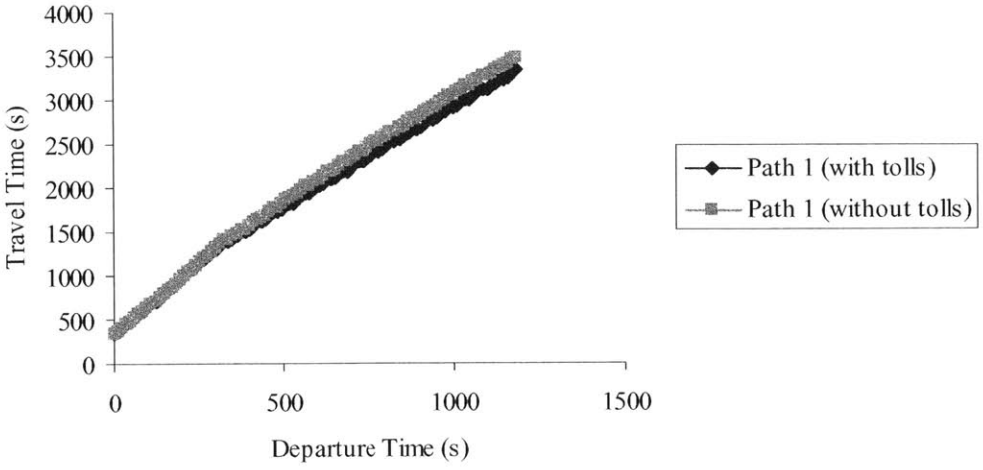


Figure 5.8. Path travel times for O-D pair a-c with and without the tolls.

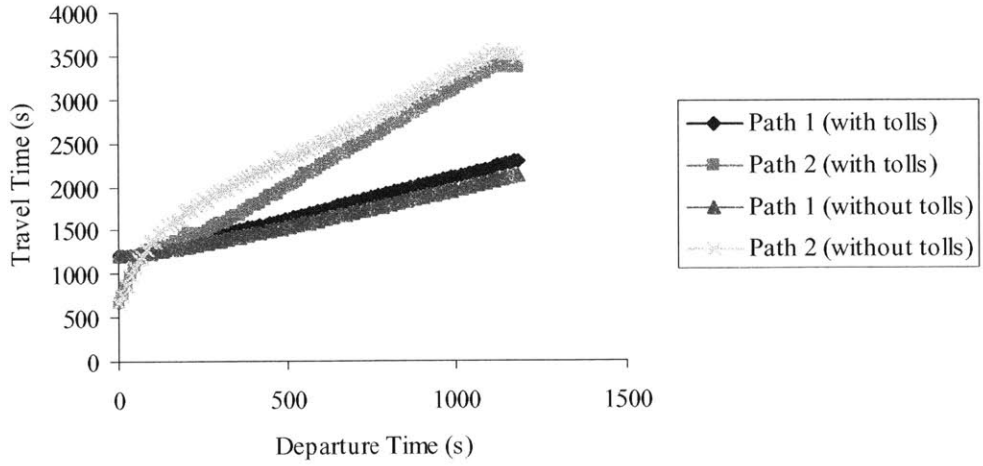


Figure 5.9. Path travel times for O-D pair b-c with and without the tolls.

5.5.2.4 Convergence of the Algorithm

The above problem has been solved using Algorithm 1 described in Section 5.2.4 because only route choice is modeled in the user behavior model within the DTA. We monitor the convergence of the algorithm by plotting the total travel time as a function of the number of iterations, as shown in Figure 5.10. The total travel time starts at 430 veh-hr and converges to 417 veh-hr after the first few iterations, thus verifying the validity of the algorithm.

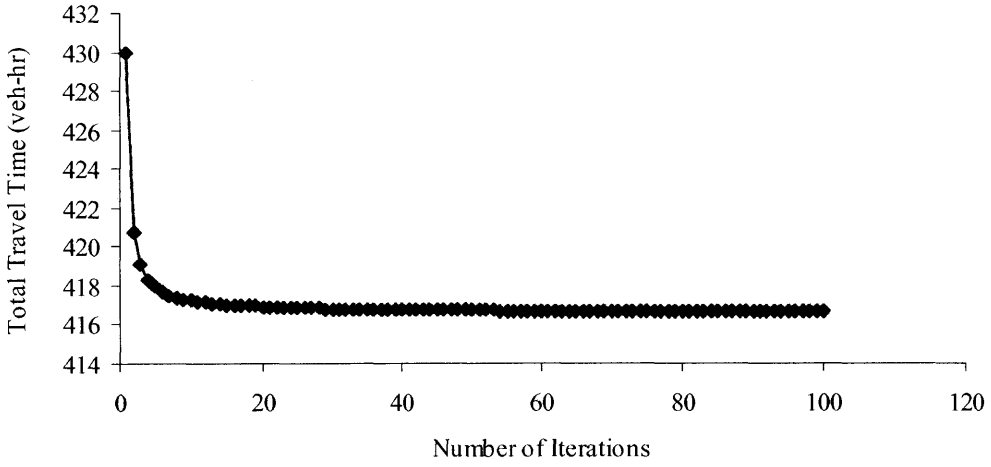


Figure 5.10. Total travel time as a function of the number of iterations used in Algorithm 1.

5.5.3 Emission Pricing with Route Choice Only: a Twelve-Link Example

5.5.3.1 Network Description

This example is taken from Xu et al. (1999). The network, shown in Figure 5.11, consists of 9 nodes, 12 links (whose attributes are shown in Table 5.4), and 7 O-D pairs (whose paths are given in Table 5.5).

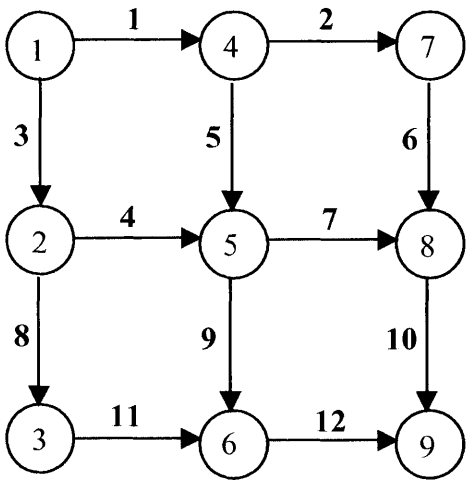


Figure 5.11. A twelve-link network example.

Table 5.4. Attributes of the links.

Link	Length (miles)	Free-Flow Speed (miles/hour)	Jam Density (vehicles/mile)
1	2.0	60	210
2	2.0	60	210
3	1.5	60	210
4	1.8	60	210
5	1.5	60	210
6	2.1	60	210
7	1.7	60	210
8	1.8	60	210
9	2.0	60	210
10	2.3	60	210
11	2.2	60	210
12	2.5	60	210

Table 5.5. O-D pairs and paths.

O-D Pair	Path
(1, 9)	1, 2, 6, 10 1, 5, 7, 10 1, 5, 9, 12 3, 4, 7, 10 3, 4, 9, 12 3, 8, 11, 12
(1, 5)	1, 5 3, 4
(5, 9)	7, 10 9, 12
(1, 3)	3, 8
(3, 9)	11, 12
(1, 7)	1, 2
(7, 9)	6, 10

5.5.3.2 Scenario Description

The demand interval is $[0,300]$ seconds. The time-dependent O-D demand is given by the following formula:

$$D^{rs}(t) = \eta^{rs}\left(\frac{t}{75} - \left(\frac{t}{150}\right)^2\right), \quad t \in [0,300]$$

where η^{rs} is given in Table 5.6.

Table 5.6. Values of the parameter η^{rs}.

O-D Pair	(1, 9)	(1, 5)	(5, 9)	(1, 3)	(3, 9)	(1, 7)	(7, 9)
η^{rs}	1.7	0.6	0.6	0.25	0.2	0.2	0.7

We assume that all users belong to the stochastic user class with a uniform value of time equal to $ 7. Moreover, we assume that there are two vehicle categories representing low and high emitters. The emission species for which the pricing is done is CO. The emission factors for the first vehicle category (low emitters) are those used in the previous example (Table 5.3). The emission factors of the high emitters are assumed to be four times the emission factors of the low emitters. Moreover, the fleet of vehicles is assumed to consist of 80 % low emitters and 20 % high emitters. The maximum toll is uniform among all links, and it is set to $ 1 and $ 2 for low and high emitters, respectively. The coefficients of the variables used in the utility functions are the same as those used in the previous example. Moreover, departure time choice is not modeled here as well.

5.5.3.3 Results

For the network and scenario shown above, emission pricing resulted in a reduction of total emissions from 9253 (50 % of which is attributable to low emitters and 50 % to high emitters) to 8645 grams (52 % of which is attributable to low emitters and 48 % to high emitters). This corresponds to a saving of 6.6 %. The savings due to low and high emitters are 28 % and 72 %, respectively. This is indicative that most of the reductions in emissions accrued from the rerouting of traffic due to pricing are attributable to high emitters, even though they represent a small proportion in the vehicle mix. Figure 5.12 shows the total CO emission levels for each O-D pair summed over all departure times with and without tolls.

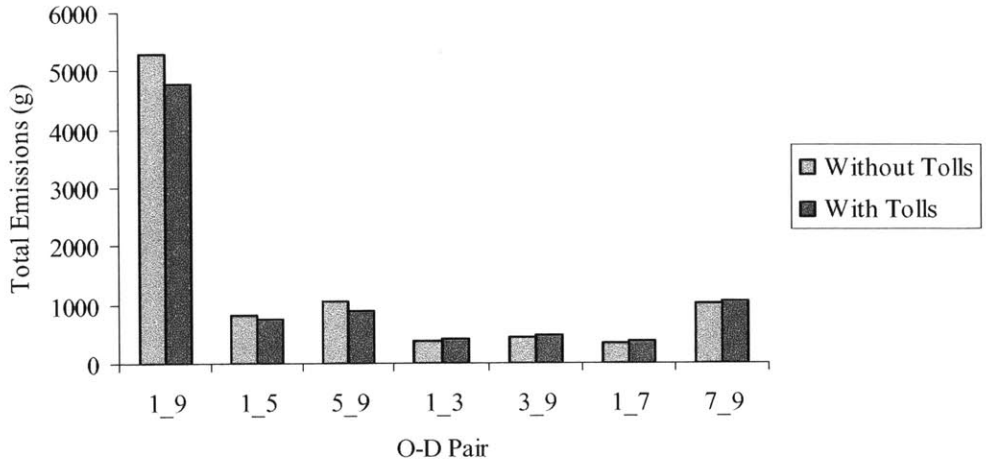
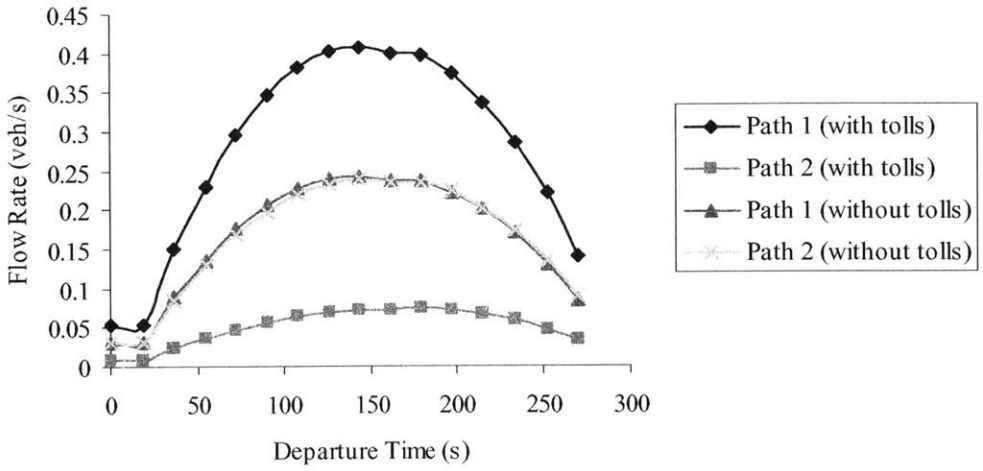


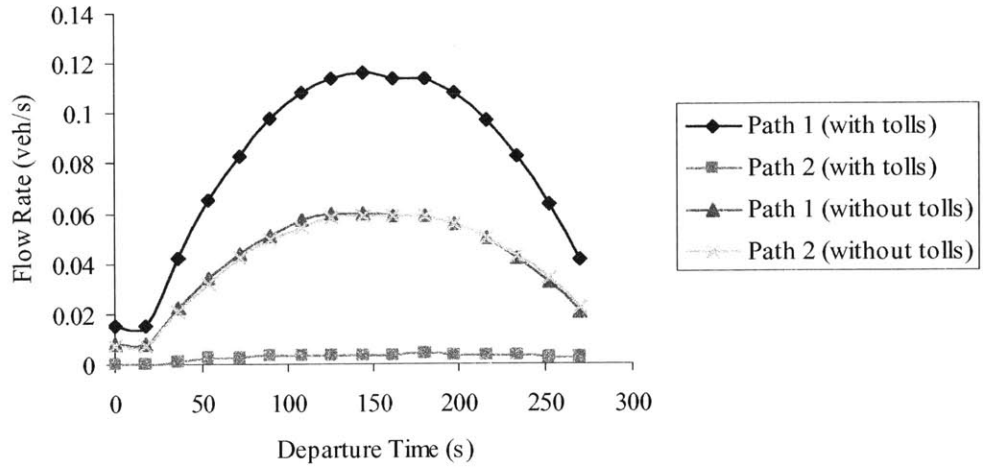
Figure 5.12. Total emissions levels by O-D pair summed over all departure times with and without tolls.

The results indicate that the emission levels for O-D pairs with multiple paths decrease whereas emissions increase for O-D pairs with only one path. This is intuitively correct since O-D pairs with multiple paths would allow the users to shift from highly polluted paths to less polluted paths where they would pay less, thus resulting in a net decrease in emissions.

Below we show the results for O-D pair 5-9 which is connected by two paths (path 1 consisting of links 7-10 and path 2 consisting of links 9-12). Figure 5.13 shows the path flow rates before and after the tolls. For both low and high emitters, both path 1 and path 2 have almost the same flow rate without the tolls since their travel times are approximately equal (see Figure 5.16). In the presence of the tolls, the flow rate on path 2 becomes very small compared to that on path 1. We explain this shift in flow rates by examining the emission levels and tolls on the two paths. Before implementing the tolls, the emissions per vehicle on path 1 are less than those on path 2 (see Figure 5.14). Thus, we expect the tolls to be higher on path 2 so as to shift vehicles to the less polluted path 1. Indeed, this is verified in Figure 5.15, where the toll on path 1 is set to zero while that on path 2 is set to the maximum. The larger flow rate on path 1 causes its travel time to increase and its emission levels per vehicle to decrease (since its average speed decreases, and the expected emission factors we are using in this example tend to increase with speed), and the smaller flow rate on path 2 causes its travel time to decrease and its emissions per vehicle to increase (see Figures 5.14 and 5.16).

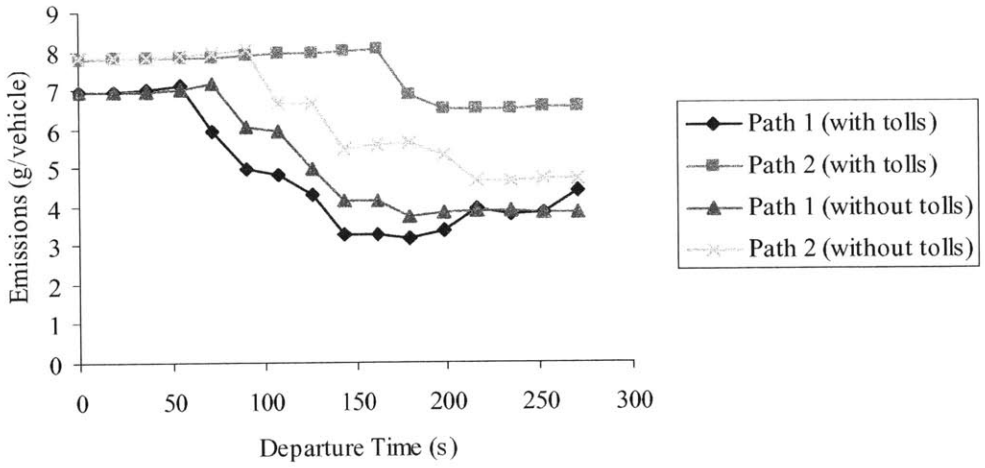


(a)

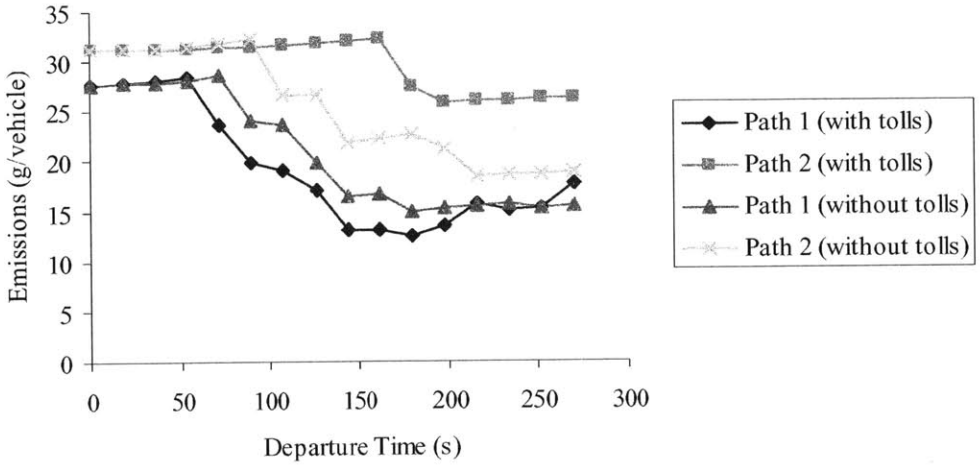


(b)

Figure 5.13. Path flow rates for O-D pair 5-9 for low emitters (part a) and high emitters (part b).

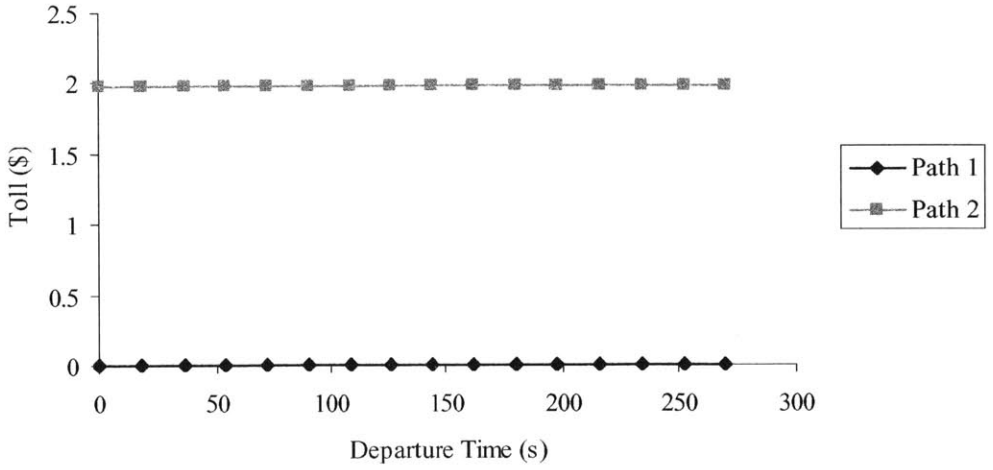


(a)

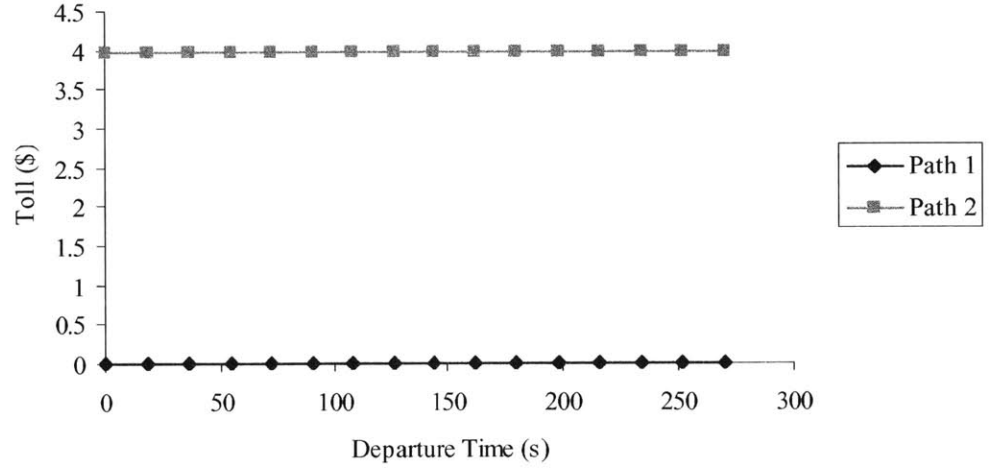


(b)

Figure 5.14. Emissions per vehicle for low emitters (part a) and high emitters (part b) on O-D pair 5-9.



(a)



(b)

Figure 5.15. Tolls for low emitters (part a) and high emitters (part b) on O-D pair 5-9.

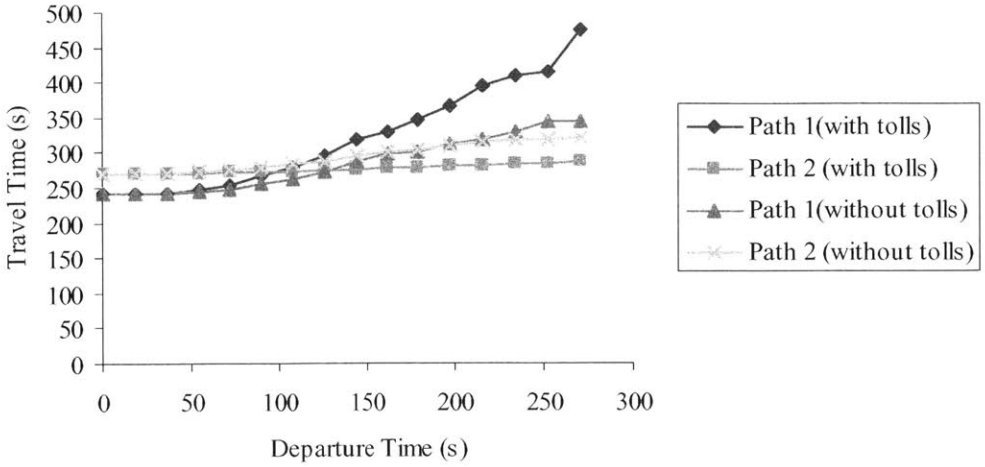


Figure 5.16. Path travel times for O-D pair 5-9.

The resulting pattern of emission prices causes the total travel time to increase by 11.3 %. This might be explained by the fact that for the pollutant used in this example, the emission factors (in g/s) increase with speed. Reductions in emissions might thus be achieved through reductions in speed, and hence increases in travel time. However, this effect might not be general since decreasing speed implies spending more time on the network, which would also generate more emissions. For the example and scenario shown above, it appears that the first effect of speed reductions dominates the second. The time-dependent tolls for all links are given in Figures 5.17- 5.28.

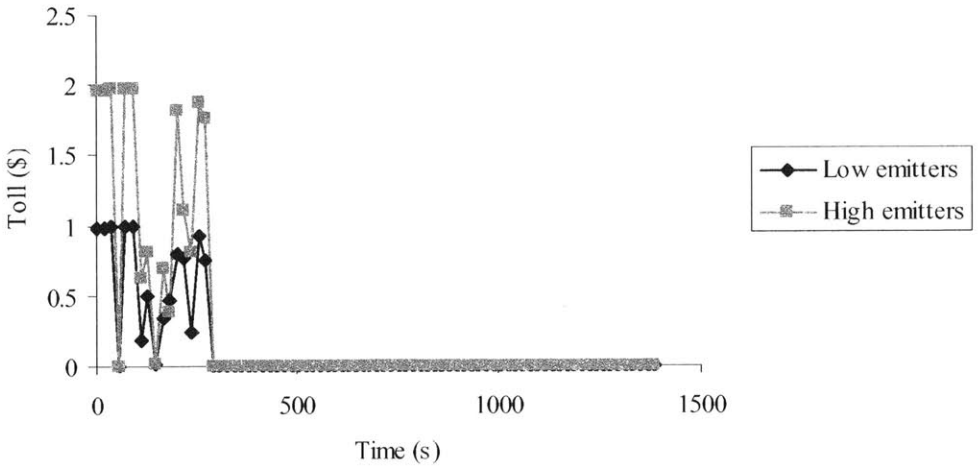


Figure 5.17. Tolls for link 1.

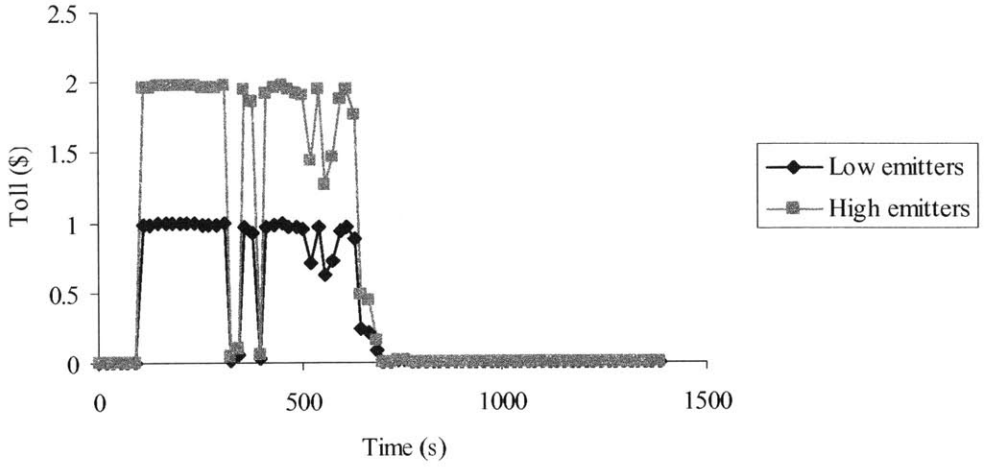


Figure 5.18. Tolls for link 2.

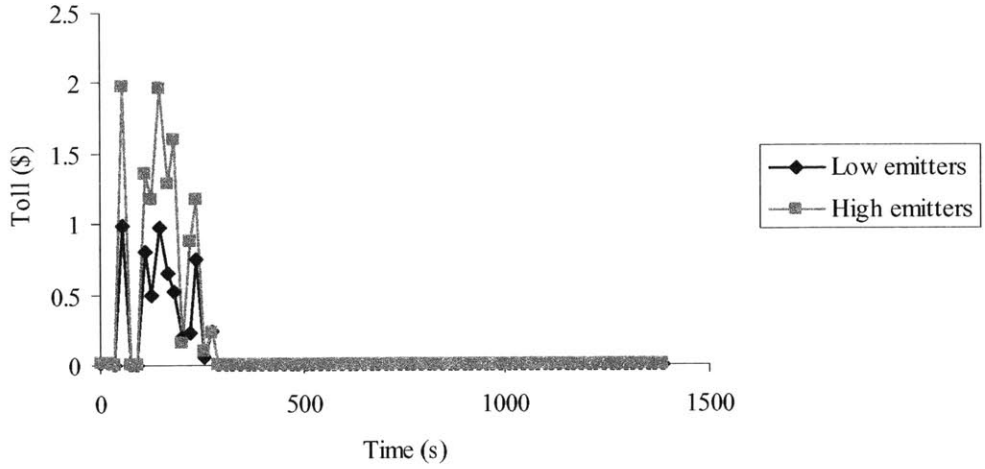


Figure 5.19. Tolls for link 3.

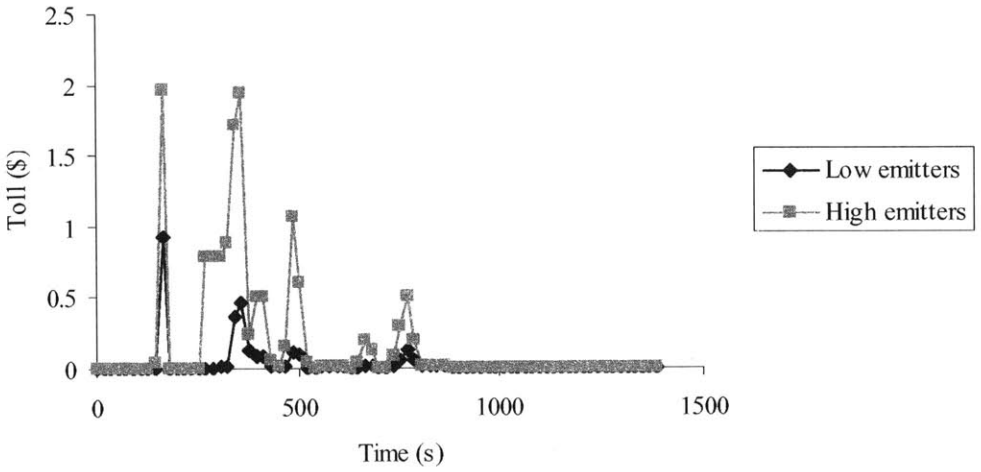


Figure 5.20. Tolls for link 4.

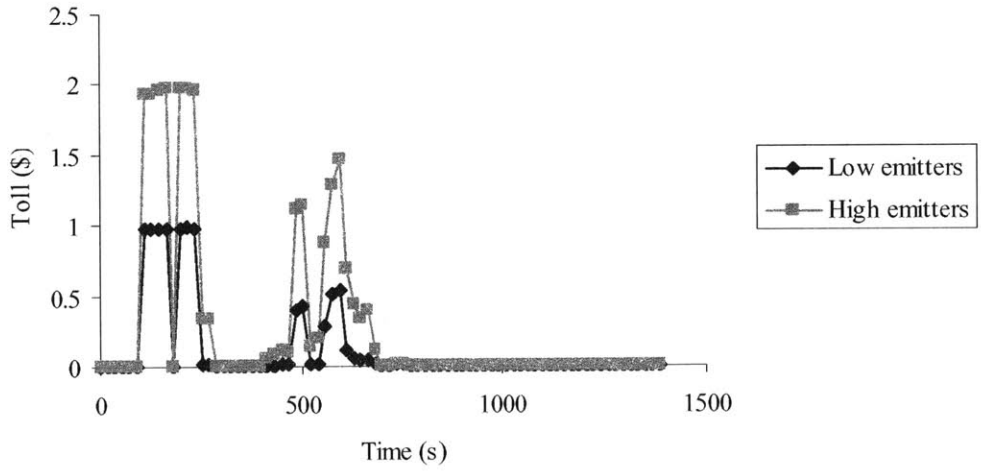


Figure 5.21. Tolls for link 5.

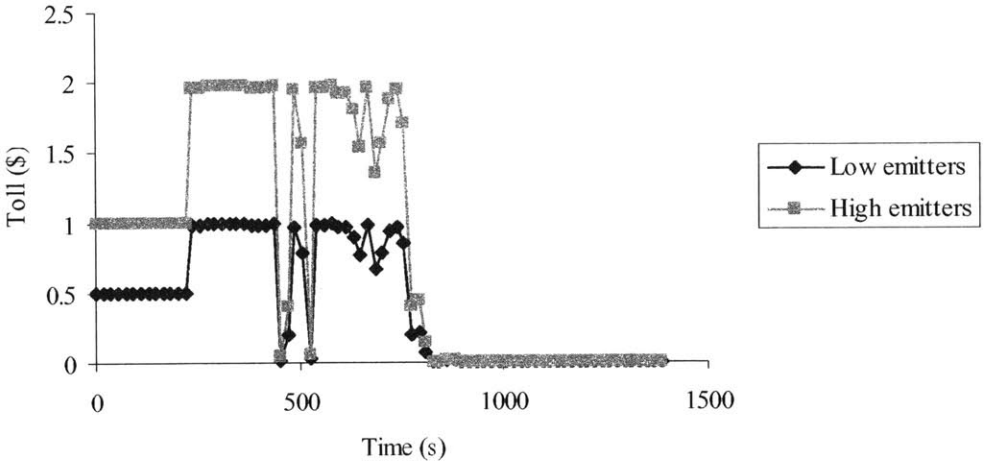


Figure 5.22. Tolls for link 6.

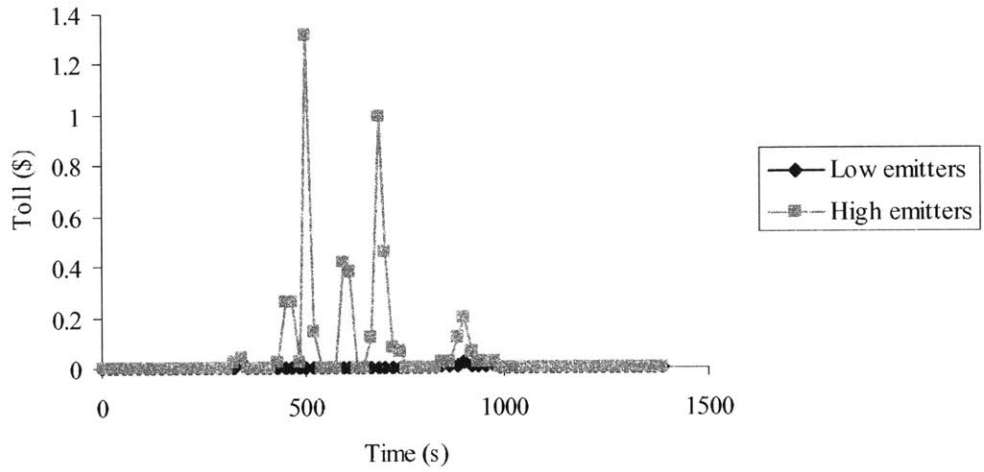


Figure 5.23. Tolls for link 7.

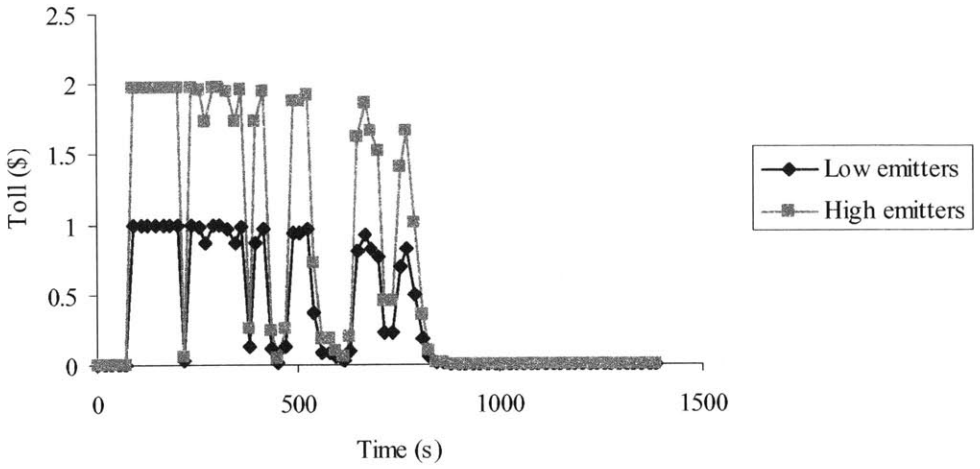


Figure 5.24. Tolls for link 8.

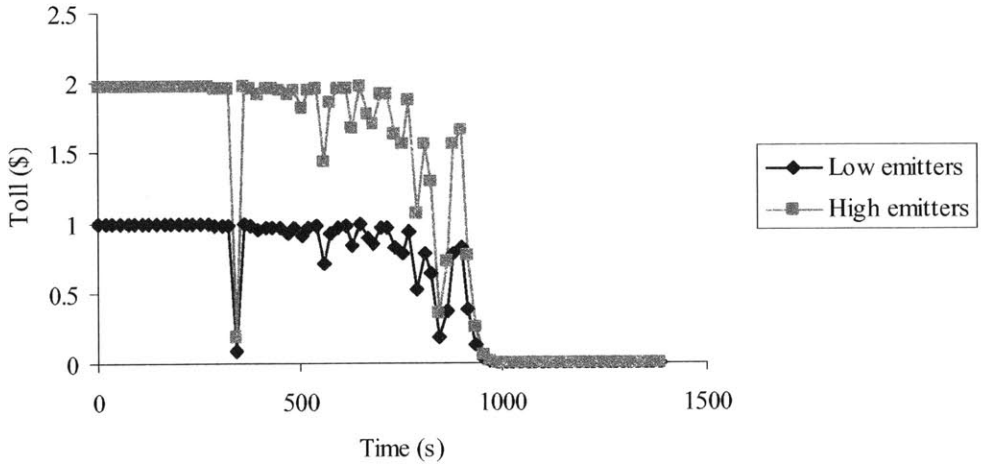


Figure 5.25. Tolls for link 9.

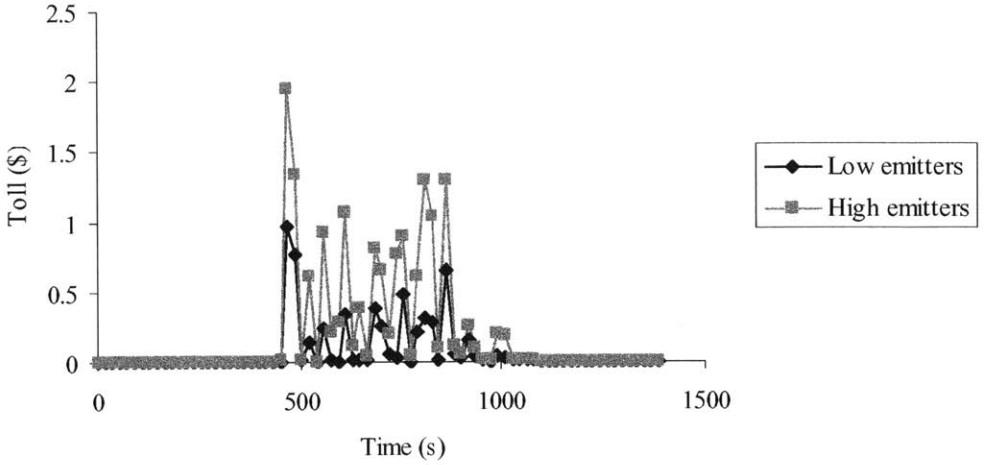


Figure 5.26. Tolls for link 10.

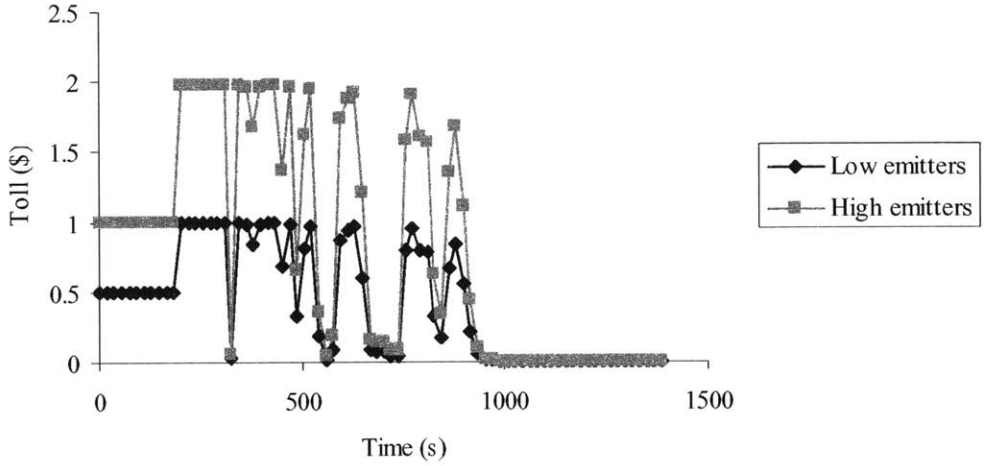


Figure 5.27. Tolls for link 11.

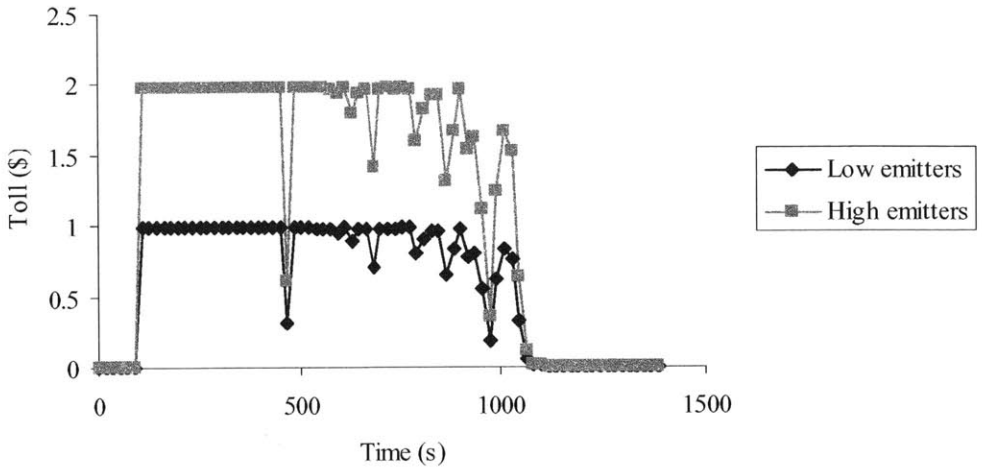


Figure 5.28. Tolls for link 12.

We have also investigated the effect of changing the maximum tolls that can be charged for every vehicle category. When the maximum toll charged for high emitters is permitted to increase from $ 2 to $ 4 while fixing the maximum toll for low emitters at $ 1 (i.e. making the ratio of maximum tolls proportional to the ratio of emission factors), the total emissions savings increased from 6.6 % to 6.8 %. When the maximum toll charged for high emitters is further increased to $ 6 (to allow for more than a proportional toll to emissions factor ratio), the total emissions savings increased to 7.5 % compared to the base case (which corresponds to maximum tolls of $ 1 and $ 2 for low and high emitters, respectively). Thus, the total emissions savings from emission pricing are sensitive to the maximum tolls that can be charged to different vehicle categories, which is a policy question that needs to be addressed.

5.5.3.4 Convergence of the Algorithm

The above problem has been solved using an adaptation of Algorithm 1 described in Section 5.2.4 (the difference is that gradient expressions and tolls are computed for each vehicle category separately) because only route choice is modeled in the user behavior model within the DTA. We monitor the convergence of the algorithm by plotting the total CO emissions as a function of the number of iterations, as shown in Figure 5.29. The figure shows that total emissions decrease as the number of iterations increases beyond

40 iterations, and oscillates afterwards in a small range (8500 to 8700 grams), which might be considered to be acceptable from a convergence standpoint.

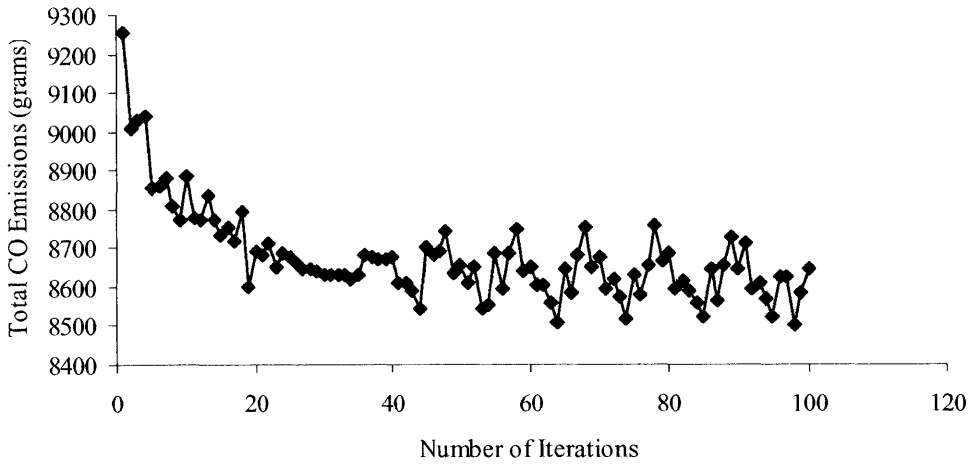


Figure 5.29. Total *CO* emissions as a function of the number of iterations used in an adaptation of Algorithm 1.

5.5.4 Congestion Pricing with Both Route and Departure Time Choices: the Twelve-Link Example

5.5.4.1 Network Description

The network used for this example is the same as that used for emission pricing in Section 5.5.3.

5.5.4.2 Scenario Description

The departure time period is $[0,3600]$ seconds. We assume that the preferred arrival time window is $[2700,3600]$ seconds for all O-D pairs. The total demand by number of vehicles for every O-D pair is shown in Table 5.7:

Table 5.7. Total O-D demand.

O-D Pair	(1, 9)	(1, 5)	(5, 9)	(1, 3)	(3, 9)	(1, 7)	(7, 9)
Demand	1400	700	700	400	250	250	400

We assume that all users belong to the stochastic user class with a uniform value of time equal to $ 7. The maximum toll is constant among all links, and it is set to $ 2. In this example, we model both route and departure time choices using a joint logit model. We use the following utility specifications:

$$\begin{cases} \widetilde{V}_m^{rs}(p) = -CF^{rs}(p) \\ \widetilde{V}_m^{rs}(t) = 0 \\ \widetilde{V}_m^{rs}(p,t) = \alpha_1\left(d^{rs}(p,t) + \frac{1}{\theta_m} q^{rs}(p,t)\right) + \alpha_2\left(t^{rs*} - \Delta^{rs} - \left(t + d^{rs}(p,t)\right)\right)^+ \\ \qquad\qquad + \alpha_3\left(t + d^{rs}(p,t) - \left(t^{rs*} + \Delta^{rs}\right)\right)^+ . \end{cases}$$

The model was not estimated in part due to lack of data. We use the coefficients estimated in Small (1982) for the disutility of travel time, early arrivals, and late arrivals: $\alpha_1 = -0.106\,\text{min}^{-1}$, $\alpha_2 = -0.065\,\text{min}^{-1}$, and $\alpha_3 = -0.254\,\text{min}^{-1}$. These coefficient values have often been used in the literature (see van Vuren et al. (1998) and Ben-Akiva et al. (1986)). In this example, we also evaluate the impacts of the congestion tolls on the total levels of tailpipe CO emissions. The emission factors that we use are shown in Table 5.3. We use Algorithm 2 to solve the congestion pricing model with route and departure time choices.

5.5.4.3 Results

The application of the time-dependent tolls to the network links for the given network parameters and user behavior models resulted in a decrease of total travel time from 472 to 402 veh-hr, corresponding to a saving of 14.8 %. As in the example presented in Section 5.5.3, we present the results for O-D pair 5-9. Figures 5.30 and 5.31 show the path flows and travel times with and without the tolls. Without the tolls, travel times are almost constant because the network is not very congested, except for an increase in travel times in the departure time period extending from 2358 to 3240 seconds. This corresponds to an increase in flow in the period $[2358,3240]$, since departures in this period lead to arrivals within the PAT window. To decrease total travel times, the total tolls levied on the paths of O-D pair 5-9 are largest in the period $[2358,3240]$, as shown

in Figure 5.32. This results in a more even spreading of the flows among the departure time intervals, as seen in Figure 5.30, and almost constant travel times, as seen in Figure 5.31.

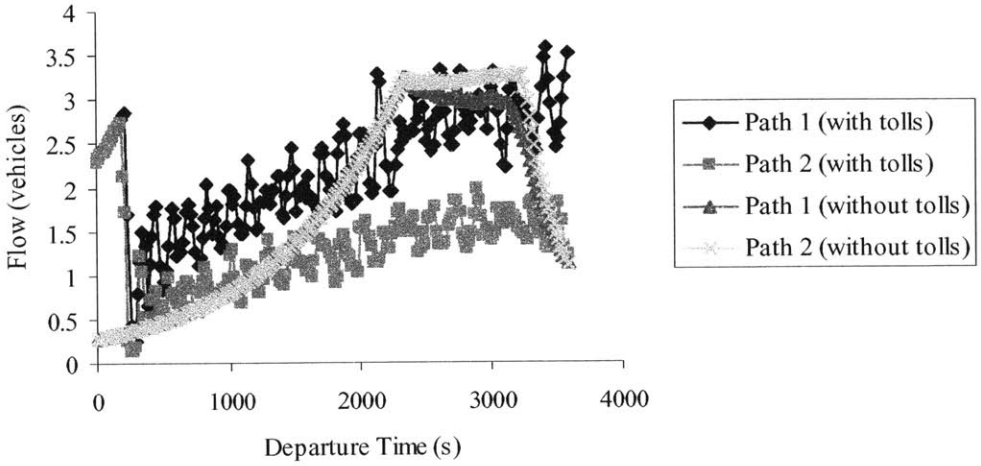


Figure 5.30. Path flows for O-D pair 5-9.

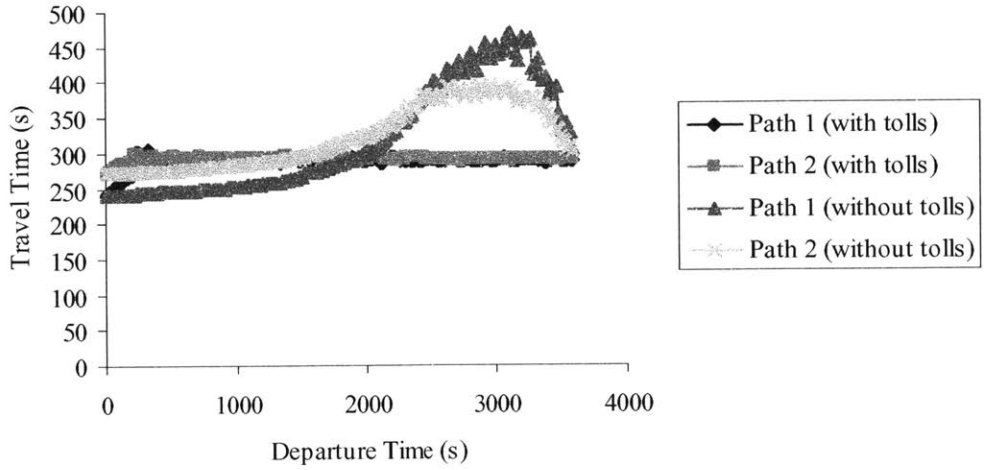


Figure 5.31. Path travel times for O-D pair 5-9.

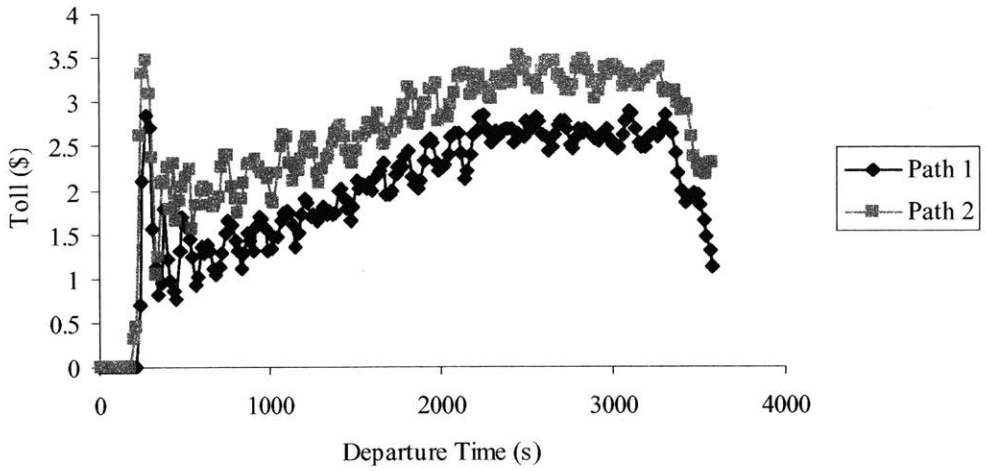


Figure 5.32. Path tolls for O-D pair 5-9.

As to the effect of the congestion tolls on total network emissions, we observe a similar effect to that in the first congestion example presented, namely a 4.4 % increase in total emissions from 25,925 to 27,059 grams of CO.

5.5.4.4 Convergence of the Algorithm

As mentioned previously, the above problem has been solved using Algorithm 2 described in Section 5.2.4. We monitor the convergence of the algorithm by plotting the total travel times as a function of the number of iterations, as shown in Figure 5.33. The figure shows that total travel time decreases as the number of iterations increases and stabilizes at a value of 402 veh-hr, thus verifying the validity of Algorithm 2.

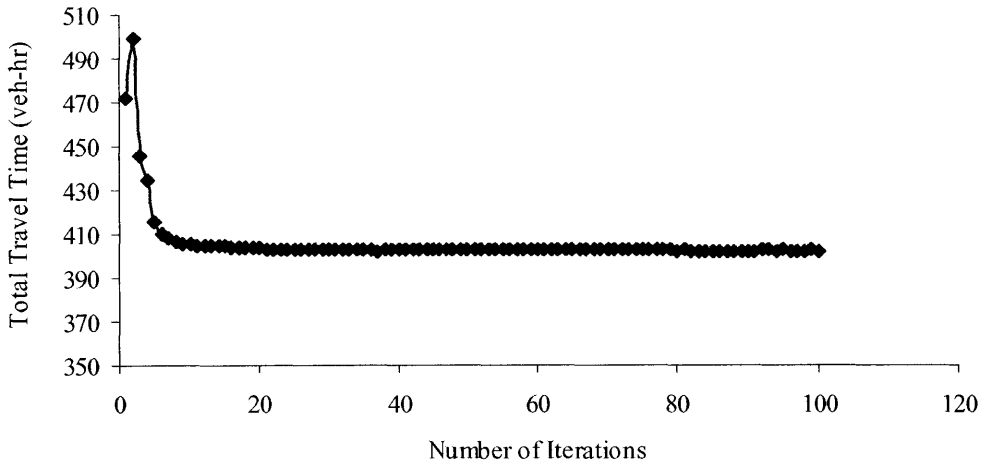


Figure 5.33. Total travel time as a function of the number of iterations used in Algorithm 2.

5.6 Conclusions, Limitations, and Extensions

In this chapter, we presented a dynamic pricing model for congestion and/or emissions in dynamic traffic networks. The pricing method is a second-best link-based approach that sets upper bounds on the link prices and takes users' reaction into account to compute the prices. The impacts of the prices on users' travel behavior have been modeled in terms of route and departure time choices. The model allows for multiple users classes distinguished for instance by value of time or vehicle category. The model has been solved using a sensitivity analysis. Gradient expressions, for the change in total travel time or emissions as a function of the change in prices, have been derived for four different behavioral assumptions: route choice only, and joint choice of route and departure time using a joint logit and two nested logit models. Iterative solution algorithms that utilize the gradient expressions to improve the solution from one iteration to another have been presented.

The framework has been extended to allow additional constraints in the model. A taxonomy of possible scenarios of interest that can arise in the combined optimization of congestion and emissions was given. Two scenarios have been formulated: the first includes an upper bound on the total emissions or total travel time, and has been formulated as a knapsack problem with arbitrary signs allowed for the values and weights

of the objects that can be included in the knapsack; the second adds a hot spot constraint to the congestion/emission pricing model, and has been formulated as a multiple knapsack problem for a simple network consisting of two parallel routes.

Various experiments were conducted on small hypothetical network examples to assess the effectiveness of the proposed pricing methods on the subsequent levels of congestion and emissions. For the network examples used and scenarios considered, congestion pricing resulted in savings in total travel times. Similarly, emission pricing (for CO) resulted in savings in total emissions. The results further indicate that decreases in total travel time do not necessarily correspond to decreases in total emissions, and vice versa, which might be the result of a trade-off between changes in speed and changes in time spent on the network. However, this effect should be studied further on other network topologies and demand scenarios.

Future research should be directed at studying how the methodology developed in this chapter can be extended or possibly modified to address the following issues:

1. The analysis in this chapter assumed that the demand is inelastic to the implemented prices. In reality, travelers might react to the prices by canceling their trips or shifting to other modes of travel such as public transit. In this context, minimizing total travel time or total emissions could be achieved by setting the tolls large enough to drive the demand to zero. As this is not practical in reality, the objective function in this case should be to maximize net social benefit. The same framework can be used to study this problem, but the gradient expressions should be modified for the new problem.
2. The developed models are useful in an offline (planning) context. If route guidance is provided to users, a route guidance generation model (see for instance Bottom (2000)) should be used instead of a dynamic traffic assignment model to find user equilibrium.
3. In the developed models, the link prices were assumed to depend on the entry time of the link. While a time-dependent link price pattern would provide a benchmark solution (in terms of least travel times or emissions), in reality travelers might be generally reluctant to unpredictable prices. It is an interesting question to study how the time-dependent price vector can be used to derive a vector of uniform (static) or stepwise tolls, that is "robust" to time-dependent changes in network conditions.

4. An interesting application of the congestion pricing model is to adapt it to a cordon pricing scheme which is a popular pricing method in practice (such as in Singapore and Norway (Emmerink (1998))). This can be done within the algorithms presented earlier by imposing zero upper bounds on prices for links that are outside the cordon and non-zero upper bounds for links that belong to the cordon.
5. The emission pricing model presented in this chapter has assumed that one emission species is being optimized. In spite of the difficulties associated with accounting for multiple emission species in the optimization procedure, the impact of the pricing on the levels of those emission species should not be ignored. One way to include multiple emission species in the model is to associate a monetary cost with a unit mass of each emission species and use these costs to transform the total emissions per vehicle into a monetary equivalent.
6. In some of the derivations of the gradient expression, it was assumed that a slight increase in a link toll has a negligible effect on the changes in path travel times and path emissions, namely that the terms $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ and $h_m^{rs*}(p,t)\frac{\partial e_m^{rs}(p,t)}{\partial q_{m'a}(l)}$ are approximately zero. While the numerical results have indicated savings in total travel times or total emissions for congestion or emission pricing, respectively, the significance of the terms $h_m^{rs*}(p,t)\frac{\partial d^{rs}(p,t)}{\partial q_a(l)}$ and $h_m^{rs*}(p,t)\frac{\partial e_m^{rs}(p,t)}{\partial q_{m'a}(l)}$ should be revisited in future research. Moreover, one has to investigate the existence of the gradients of total travel time and total emissions with respect to link tolls.
7. Finally, we make some comments on the departure time choice model. First, the discretization of the departure time alternatives in the departure time choice model was assumed to be identical to the discretization used within the DTA. If the departure time intervals are very small and network conditions do not change instantaneously, travelers might not be able to perceive differences between adjacent departure time alternatives. Thus, the models and algorithms should be modified to allow for departure time alternatives in the departure time choice model that are larger in duration than those used in the DTA. Second, different tree structures for the joint choice of route and departure time that allow for more flexibility in the error

structure could be investigated. Third, in the analysis, departure time *choice* rather than departure time *change* (which arises in an ATIS context) has been studied. It is useful to investigate whether the developed models and algorithms could be modified to model deviations from habitual departure time behavior as a response to pre-route or en-route information.

Chapter 6

Conclusions and Directions for Future Research

6.1 Contributions and Major Results

The major theme of this thesis has been the development of models and algorithms for the optimization of traffic flows and emissions in dynamic traffic networks. The developed methods fall in two categories: methods for the enhanced representation of traffic flows and emissions, and methods for their management via routing and pricing. Below we summarize the main contributions of this thesis as well as major results that have been obtained from this research.

In Chapter 2, a probabilistic approach was developed to model acceleration in traffic networks as a random variable that is a function of speed and road type. The approach was applied to trip data collected in South Eastern Michigan. For every speed range and road type, an acceleration distribution and a deceleration distribution were plotted. Half-normal distributions were fitted to the statistical sample distributions, and the goodness-of-fit was shown to be acceptable in most cases, justifying the validity of the probabilistic approach. The standard deviation of the distributions was shown to decrease as the speed range increases, and little variation was seen among road types. The acceleration model was used in conjunction with an instantaneous emission model to model emissions as

random variables and generate expected emission factors. The model thus has applications related to the integration of non-microscopic dynamic traffic models, that generate speed but not acceleration as output, and instantaneous emission models that require both speed and acceleration as input.

Routing algorithms, which arise as sub-problems in dynamic traffic assignment models as well as in other applications, were developed in Chapters 3 and 4 of the thesis. Specifically, in Chapter 3, we studied the minimum cost flow problem in capacitated dynamic networks where a given supply should be sent from an origin node to a destination node at a certain departure time in minimum cost while satisfying the link capacities and assuming no waiting is allowed. We used the well-known successive shortest path algorithm to solve the problem. We developed two algorithms, denoted as Algorithm B and Algorithm C, for the shortest path computation involved in the solution of the dynamic minimum cost flow problem. We also reviewed an algorithm, due to Cai et al. (2001), which we denoted as Algorithm A. Algorithms A, B, and C were implemented, and their computational efficiencies were assessed by using large-size capacitated dynamic networks. The computational results indicated that Algorithms B and C are more efficient than Algorithm A. Moreover, for the test networks used, the successive shortest path algorithm employing Algorithm C achieved significant time savings, compared to that employing Algorithm A (by up to a factor of 113) and that employing Algorithm B (by up to factors of 25, 39, and 72 for three different implementations of Algorithm B). We extended the analysis to study the case of the minimum travel time problem, which is a special case of the minimum cost flow problem, but with additional properties that could be exploited in the solution algorithms. We also discussed the cases of waiting, multiple origins, multiple destinations, and multiple departure times.

In Chapter 4, we conducted experimental analyses of a new approach developed in Chabini (2002) for solving the shortest path problem in static and dynamic First-In-First-Out (FIFO) networks. The new algorithm is similar in terms of the basic steps to label-setting comparison-based algorithms, such as Dijkstra's algorithm, but tries to reduce the number of nodes that need to be sorted, which is the bottleneck operation in Dijkstra's algorithm. This is done by introducing optimality conditions that detect whether a node

has been permanently set, in which case the node does not get inserted in the heap, and by delaying the entry to the heap of those nodes that are not yet label set hoping that they would satisfy the optimality conditions at a later stage of the algorithm. The numerical results indicated that a significant percentage of nodes are known to be optimal without entry to the heap, and that this percentage increases as the optimality conditions become stronger.

In Chapter 5, we studied dynamic pricing methods for congestion and emissions in dynamic traffic networks. We formulated a dynamic link-based second-best congestion pricing model as a bi-level program, where the upper level is to minimize total travel time subject to upper bounds on the link prices and the lower level is the users' reaction function (different user classes were allowed). We studied the analytical properties of the model and used a sensitivity analysis method to solve it. We derived gradient expressions for the change in total travel time as a function of the change in prices under two assumptions: users react to the prices by adjusting their route choice only, and users react to the prices by adjusting both their route and departure time choices. We used the gradient expressions in the development of iterative solution algorithms to solve the model. We extended the congestion pricing model and algorithms to study dynamic emission pricing, where the prices vary also by vehicle category. Finally, we provided a taxonomy of emission or congestion-related constraints that could be included in the model. We formulated two variants: congestion (emission) pricing subject to constraints on the total emissions generated (total travel time), and congestion/emission pricing subject to hot spot environmental constraints. The experimental analyses conducted on small hypothetical network examples indicate that the pricing methods could achieve reasonable reductions in the criterion (total travel time or emissions) being optimized, and that the effect on the other criterion is not always positive or negative.

6.2 Directions for Future Research

In this section, we summarize directions for future research as related to each chapter of this thesis.

For the probabilistic acceleration modeling approach in Chapter 2, it would be useful to investigate the nature of the fitted acceleration and deceleration distributions and their

variation with road type by applying the developed methodology to other data sets (namely the Sierra chase car data). It would also be important to quantify the activity from freeway ramps. Moreover, it would be interesting to investigate the effects of driver aggressiveness and vehicle type on the variation of acceleration and deceleration distributions.

For the dynamic minimum cost flow problem studied in Chapter 3, one direction for future research is to evaluate the empirical performance of the developed algorithms when applied to cases involving waiting and/or multiple origins, destinations, and departure times. Particularly, for Algorithm C, it would be interesting to investigate to what extent the lower bounds utilized in the algorithm remain effective when the number of destinations with positive demand increases. If this latter number is close to n, then Algorithms B and C would have the same performance.

The new approach for solving shortest path problems described in Chapter 4 leads to a promising avenue of research, namely how one can detect the optimality of node labels without the need for costly sorting operations used in traditional comparison-based label-setting algorithms. The use of optimality conditions, other than those used in Chapter 4, and the development of enhanced implementations of the algorithms that could result in more savings in running times are left for future work. The latter is especially important given the large percentage of nodes that are detected to be optimal by the algorithm and are thus saved entry to the heap.

For the dynamic congestion/emission pricing methods developed in Chapter 5, several directions can be identified for extending or modifying the basic framework. Regarding the tolls, it would be useful to apply the developed methodology to a cordon pricing scheme which is a common pricing method in practice. Moreover, since travelers might be uncomfortable with time-dependent tolls or since those tolls might be difficult to enforce in practice, one future research direction is to investigate whether one could derive a static or step toll pattern that is "robust" to changes in network conditions. Regarding the model of the joint choice of route and departure time used to predict users' travel adjustment to the levied tolls, behavioral models other than joint and nested logit could be investigated; the departure time choice model should be modified to allow for different discretizations in departure time alternatives than the discretizations used for the

assignment; and the analysis could be extended to model departure time change rather than departure time choice, as has been assumed in this thesis. Finally, other extensions include relaxing the assumption of inelastic demand, including the effect of information, and extending the emission pricing model to optimize the levels of multiple emission species.

References

Abkowitz, M. (1980). The impact of service reliability on work travel behavior, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Abou Zeid M., I. Chabini, E. K. Nam, and A. Cappiello (2002). Probabilistic modeling of acceleration in traffic networks as a function of speed and road type. Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems, Singapore, September 2002, pp. 472-478.

Abou Zeid, M., and I. Chabini (2002). Combined mobility and emissions problems in dynamic traffic networks: taxonomy, formulations, and solution algorithms. Internal paper, Massachusetts Institute of Technology.

Agneiv, C. E. (1977). The theory of congestion tolls. *Journal of Regional Science,* Vol. 17, pp. 381-393.

Ahmed, K. I. (1999). Modeling drivers' acceleration and lane changing behavior. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Ahuja, R., T. Magnanti, and J. Orlin (1993). *Network flows: theory, algorithms, and applications.* Prentice Hall, Englewood Cliffs, NJ.

Alfa, A. S. (1986). A review of models for the temporal distribution of peak traffic demand. *Transportation Research B*, Vol. 20B, No. 6, pp. 491-499.

Antoniou, C. (1997). Demand simulation for dynamic traffic assignment. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Arnott, R., A. de Palma, and R. Lindsey (1990). Departure time and route choice for the morning commute. *Transportation Research B*, Vol. 24B, No. 3, pp. 209-228.

Barratt R. (2001). *Atmospheric dispersion modelling: an introduction to practical applications.* Earthscan Publications, London, United Kingdom.

Barth, M. (1998). Integrating a Modal Emissions Model into Various Transportation Modeling Frameworks. *Transportation Planning and Air Quality III, Emerging Strategies and Working Solutions* (S. Washington, ed.), ASCE, New York, pp. 38-50.

Bates, J. J. (1996). Time period choice modeling: a preliminary review. Final report for the Department of Transport, John Bates Services, Oxford.

Ben-Akiva, M., and M. Bierlaire (1999). Discrete choice methods and their applications to short term travel decisions. In R. Hall (ed.), *Handbook of Transportation Science*, International Series in Operations Research and Management Science, Vol. 23, Kluwer. http://roso.epfl.ch/mbi/handbook-final.pdf.

Ben-Akiva, M., A. de Palma, and P. Kanaroglou. (1986). Dynamic model of peak period traffic congestion with elastic arrival rates. *Transportation Science*, Vol. 20, No. 2, pp. 164-181.

Ben-Akiva, M., and S. R. Lerman (1979). Disaggregate travel and mobility choice models and measures of accessibility. In *Behavioral travel modeling*. D. Hensher and P. Stopher, eds. Croom Helm, London.

Ben-Akiva, M., and S. R. Lerman (1985). *Discrete choice analysis*. The MIT Press, Cambridge, Massachusetts.

Bertsimas, D., and S. Stock Patterson (1998). The air traffic flow management problem with enroute capacities. *Operations Research*, Vol. 46, pp. 406-422.

Bottom, J. A. (2000). Consistent Anticipatory Route Guidance. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Braid, R. M. (1996). Peak-load pricing of a transportation route with an unpriced substitute. *Journal of Urban Economics,* Vol. 40, pp. 179-197.

Brotcorne, L., M. Labbé, P. Marcotte, and G. Savard (2001). A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science,* Vol. 35, No. 4, pp. 345-358.

Cai, X., T. Kloks, and C. K. Wong (1997). Time-varying shortest path problems with constraints. *Networks* 29 (3), pp. 141-149.

Cai, X., D. Sha, and C. K. Wong (2001). Time-varying minimum cost flow-problems. *European journal of operational research*, Vol. 131, pp. 352-374.

Cappiello, A. (2002). Modeling traffic flow emissions. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Cappiello A., I. Chabini, E. K. Nam, A. Luè, and M. Abou Zeid (2002). A statistical model of vehicle emissions and fuel consumption. Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems, Singapore, September 2002, pp. 801-809.

Carey, M., and A. Srinivasan (1993). Externalities, average and marginal costs, and tolls on congested networks with time-varying flows. *Operations Research,* Vol. 41, pp. 217-231.

Carlson, T. R., and T. C. Austin (1997). Development of speed correction cycles. Draft report prepared for the Environmental Protection Agency.

Cascetta, E. (2001). *Transportation systems engineering: theory and methods*. Kluwer Academic Publishers, Dordrecht, Boston, Massachusetts.

Cascetta, E., A. Nuzzolo, and L. Biggiero (1992). Analysis and modeling of commuters' departures and route choices in urban networks. *Proceedings of the Second International CAPRI Seminar on Urban Traffic Networks*.

Cascetta, E., A. Nuzzolo, F. Russo, and A. Vitetta (1996). A modified logit route choice model overcoming path overlapping problems. Specification and some calibration results for interurban networks. *Proceedings of the 13th International Symposium on the Theory of Road Traffic Flow*, Lyon, France.

Chabini, I. (1998). Discrete dynamic shortest path problems in transportation applications: complexity and algorithms with optimal run time. *Transportation Research Record* 1645, pp. 170-175.

Chabini, I. (2002). A note on improved shortest path algorithms. Massachusetts Institute of Technology.

Chabini, I., and M. Abou Zeid (2002). The minimum travel time flow problem in capacitated dynamic networks. Internal paper, Massachusetts Institute of Technology.

Chabini, I., and M. Abou Zeid (2003). The minimum cost flow problem in capacitated dynamic networks. Presented at the 82nd annual meeting of the Transportation Research Board, Washington, D.C.

Chabini, I., and B. Dean (1998). Shortest path problems in deterministic discrete-time dynamic networks: complexity, algorithms and efficient implementations. Internal paper, Massachusetts Institute of Technology.

Chabini, I., and S. Lan (2002). Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 3, Issue 1, pp. 60-74.

Clegg, J., M. Smith, Y. Xiang, and R. Yarrow (2001). Bilevel programming applied to optimising urban transportation. *Transportation Research B,* Vol. 35, pp. 41-70.

Cooke, L., and E. Halsey (1966). The shortest route through a network with time-dependent internodal transit times. *Journal of Mathematical Analysis and Applications,* Vol. 14, pp. 492-498.

Cosslett, S. (1977). In *Demand model estimation and validation*, Vol. V, *Urban Travel Demand Forecasting Project*, D. McFadden et al. (eds.), Institute of Transportation Studies, University of California, Berkeley.

Dafermos, S., and F. T. Sparrow (1971). Optimal resource allocation and toll patterns in user-optimized transport networks. *Journal of Transport Economics and Policy,* Vol. 5, pp. 184-200.

Deakin, E., and G. Harvey (1996). Transportation pricing strategies for California: an assessment of congestion, emissions, energy, and equity impacts. Final report for California Environmental Protection Agency, Air Resources Board. http://www.arb.ca.gov/research/abstracts/92-316.htm#Abstract.

Deakin, E., and G. Harvey (1998). Transportation pricing strategies for California: an assessment of congestion, emissions, energy, and equity impacts. Research notes. Available on-line: http://www.arb.ca.gov/research/resnotes/notes/98-1.htm.

Delucchi, M. A. (2000). Environmental externalities of motor-vehicle use in the US. *Journal of Transport Economics and Policy*, Vol. 34, Part 2, pp. 135-168.

De Palma, A., M. Ben-Akiva, C. Lefèvre, and N. Lithinas (1983). Stochastic equilibrium model of peak period traffic congestion. *Transportation Science*, Vol. 17, pp. 430-453.

Downs, A. (1992). *Stuck in traffic: coping with peak-hour traffic congestion.* The Brookings Institution, Washington, D.C., and the Lincoln Institute of Land Policy, Cambridge, Massachusetts.

Dreyfus, S. E. (1969). An appraisal of some shortest-path algorithms. *Operations Research*, Vol. 17, pp. 395-412.

Emmerink, R. H. M. (1998). *Information and pricing in road transportation.* Springer-Verlag, New York.

Environmental Protection Agency, Regional and State Programs Division, Office of Mobile Sources. (1997). Opportunities to improve air quality through transportation pricing programs. http://www.epa.gov/otaq/market/pricing.pdf.

Environmental Protection Agency (1998). Technical methods for analyzing pricing measures to reduce transportation emissions. EPA 231-R-98-006, Washington, D.C. http://www.epa.gov/otaq/transp/anpricng.pdf.

Eskeland, G. S., and S. Devarajan (1996). Taxing bads by taxing goods: pollution control with presumptive charges. The International Bank for Reconstruction and Development / The World Bank, Washington, D. C.

Fancher P., R. Ervin, J. Sayer, M. Hagan, S. Bogard, Z. Bareket, M. Mefford, and J. Haugen (1998). *Intelligent cruise control field operational test.* Final Report, volume 1, United States Department of Transportation.

Gomez-Ibanez, J, and K. Small (1994). NCHRP synthesis 210 of highway practice: road pricing for congestion management: a survey of international practice. *Rep.*, National Research Council, Washington, D.C.

Gopalan, R., K. Kolluri, R. Batta, and M. Karwan (1990). Modeling equity of risk in the transportation of hazardous materials. *Operations Research*, Vol. 38, pp. 961-973.

Grier N., and I. Chabini (2002). A new approach to compute minimum time path trees in FIFO time dependent networks. *Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems*, Singapore, pp. 485-490.

Guensler, R. S., S. Washington, and W. Bachman (1998). Overview of the MEASURE modeling framework. *Transportation Planning and Air Quality III, Emerging Strategies and Working Solutions* (S. Washington, ed.), ASCE, New York, pp. 51-70.

Hague Consulting Group, Halcrow Fox, and Imperial College (2002). Modelling peak spreading and trip retiming – phase II. Final report for the Department of the Environment, Transport and the Regions, Cambridge.

Harrington, W., V. McConnell, and A. Alberini (1996). Economic incentive policies under uncertainty: the case of vehicle emission fees. Resources for the Future, discussion paper 96-32, Washington, D.C. http://www.rff.org/CFDOCS/disc_papers/PDF_files/9632.pdf.

Harrington, W., M. Walls, and V. McConnell (1995). Driving our way to cleaner air. *Issues in Science and Technology*, Winter 1994/1995.

He, Y. (1997). A flow-based approach to the dynamic traffic assignment problem: formulations, algorithms and computer implementations. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Henderson, J. V. (1974). Road congestion: a reconsideration of pricing theory. *Journal of Urban Economics,* Vol. 1, pp. 346-365.

Hendrickson, C., and G. Kocur. (1981). Schedule delay and departure time decisions in a deterministic model. *Transportation Science*, Vol. 15, No. 1, pp. 62-77.

Hoogendoorn, S. P., and P. H. L. Bovy (2001). State-of-the-art of vehicular traffic flow modelling. *Journal of Systems and Control Engineering,* special issue on road traffic modelling and control, Vol. 215, No. 4. http://cttrailf.ct.tudelft.nl/T&E/papers_course_IV_9/state-of-the-art.PDF.

Huang, H. J., and H. Yang (1996). Optimal variable road-use pricing on a congested network of parallel routes with elastic demand. *Proceedings of the 13th International Symposium on the Theory of Traffic Flow and Transportation,* pp. 479-500.

Hyman, G. (1997). The development of operational models for time period choice. Department of the Environment, Transport and the Regions, HETA Division, London.

Kaufman, D. E., and R. L. Smith (1993). Fastest paths in time-dependent networks for intelligent-vehicle-highway systems application. *IVHS Journal*, Vol. 1, pp. 1-11.

Kessler, J., and W. Schroeer (1993). Meeting mobility and air quality goals: strategies that work. U.S. Environmental Protection Agency, Office of Policy Analysis.

Knight, F. H. (1924). Some fallacies in the interpretation of social cost. *Quarterly Journal of Economics,* Vol. 38, pp. 582-606.

Labbé, M., P. Marcotte, and G. Savard (1998). A bilevel model of taxation and its application to optimal highway pricing. *Management Science,* Vol. 44, No. 12, Part 1 of 2, pp. 1608-1622.

LeBlanc, D. C., F. M. Saunders, M. D. Meyer, and R. Guensler (1995). Driving pattern variability and impacts on vehicle carbon monoxide emissions. *Transportation Research Record* 1472, pp. 45-52.

Lévy-Lambert, H. (1968). Tarification des services à qualité variable: application aux péages de circulation. *Econometrica,* Vol. 36 (3-4), pp. 564-574.

Liu, L. N., and D. E. Boyce (2002). Variational inequality formulation of the system-optimal travel choice problem and efficient congestion tolls for a general transportation network with multiple time periods. *Regional Science and Urban Economics,* Vol. 32, pp. 627-650.

Liu, Y., and H. S. Mahmassani (1998). Dynamic aspects of departure time and route decision behavior under ATIS: modeling framework and experimental results. Presented at the 77th annual meeting of the Transportation Research Board, Washington, D.C.

Liu, L. N., and J. F. McDonald (1998). Efficient congestion tolls in the presence of unpriced congestion: a peak and off-peak simulation model. *Journal of Urban Economics,* Vol. 44, pp. 352-356.

Liu, L. N., and J. F. McDonald (1999). Economic efficiency of second-best congestion pricing schemes in urban highway systems. *Transportation Research B,* Vol. 33, pp. 157-188.

Liu, T. G. (2003). Modeling and experimental studies of network traffic emissions using a microscopic simulation approach. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Lo, H. K., and M. D. Hickman (1997). Toward an evaluation framework for road pricing. *Journal of Transportation Engineering*, Vol. 123, No. 4, pp. 316-324.

Mahmassani, H. S., and G. L. Chang (1985). Dynamic aspects of departure time choice behavior in a commuting system: theoretical framework and experimental analysis. *Transportation Research Record* 1037, pp. 88-101.

Mahmassani, H. S., and G. L. Chang (1986). Specification and estimation of a dynamic departure time acceptability model in urban commuting. Presented at the 65th annual meeting of the Transportation Research Board.

Mahmassani, H. S., and G. L. Chang (1987). On boundedly rational user equilibrium in transportation systems. *Transportation Science*, Vol. 21, pp. 89-99.

Mahmassani, H. S., and R. Herman (1984). Dynamic user equilibrium departure time and route choice on idealized traffic arterials. *Transportation Science*, Vol. 18, No. 4, pp. 362-384.

Marchand, M. (1968). A note on optimal tolls in an imperfect environment. *Econometrica*, Vol. 36 (3-4), pp. 575-581.

McDonald, J. F., E. L. d'Ouville, and L. N. Liu (1999). *Economics of urban highway congestion and pricing*, Kluwer Academic Publishers.

McFadden, D. (1978). Modelling the choice of residential location. In *Spatial interaction theory and residential location*. A. Karlquist et al., eds. North Holland, Amsterdam, pp. 75-96.

Miller-Hooks, E. (corresponding author), and S. Stock Patterson (2002). On solving quickest time problems in time-dependent, dynamic networks. Unpublished data. Corresponding author: Department of Civil and Environmental Engineering, the Pennsylvania State University, e-mail: edm3@psu.edu.

Nagurney, A. (2000 a). Congested urban transportation networks and emission paradoxes. *Transportation Research Part D* 5, pp. 145-151.

Nagurney, A. (2000 b). *Sustainable transportation networks*. Edward Elgar Publishing, Inc., Massachusetts.

National Research Council Committee to Review EPA's Mobile Source Emissions Factor (MOBILE) Model (2000). *Modeling mobile-source emissions*. National Academy Press.

Orda, A., and R. Rom (1990). Shortest-path and minimum-delay algorithms in networks with time-dependent edge length. *Journal of the ACM*, Vol. 37 (3), pp. 607-625.

Pallottino, S., and M. G. Scutellà (1998). Shortest path algorithms in transportation models: classical and innovative aspects. In (P. Marcotte and S. Nguyen, eds.) *Equilibrium and Advanced Transportation Modelling*, Kluwer, pp. 245-281.

Patriksson, M., and R. T. Rockafellar (2002). A mathematical model and descent algorithm for bilevel traffic management. *Transportation Science,* Vol. 36, No. 3, pp. 271-291.

Pigou, A. C. (1920). *Wealth and welfare.* Macmillan, London.

Ran, B., R. W. Hall, and D. E. Boyce (1996). A link-based variational inequality model for dynamic departure time/route choice. *Transportation Research B*, Vol. 30, No. 1, pp. 31-46.

Roberts, C. A., S. Washington, and J. D. Leonard II (1999). Forecasting dynamic vehicular activity on freeways: bridging the gap between travel demand and emerging emissions models. *Transportation Research Record* 1664, pp. 31-39.

Small, K. A. (1982). The scheduling of consumer activities: work trips. *The American Economic Review*, Volume 72, Issue 3, pp. 467-479.

Small, K. A. (1987). A discrete choice model for ordered alternatives. *Econometrica* 55 (2), pp. 409-424.

Small, K. A. (1992 a). Trip scheduling in urban transportation analysis. *The American Economic Review*, Vol. 82, Issue 2, pp. 482-486.

Small, K. A. (1992 b). *Urban Transportation Economics*. Harwood Academic Publishers, Chur, Switzerland.

Special Report 209: Highway Capacity Manual, 3rd ed. (1998). TRB, National Research Council, Washington, D.C.

van Vuren, T., A. Daly, and G. Hyman (1998). Modelling departure time choice. Colloquium Vervoersplanologisch Speurwerk, Amsterdam. http://www.contram.com/TECH/PAPER02.HTM.

Verhoef, E. T. (2002). Second-best congestion pricing in general static transportation networks with elastic demands. *Regional Science and Urban Economics,* Vol. 32, pp. 281-310.

Verhoef, E. T., P. Nijkamp, and P. Rietveld (1996). Second-best congestion pricing: the case of an untolled alternative. *Journal of Urban Economics,* Vol. 40 (3), pp. 279-302.

Vickrey, W. (1969). Congestion theory and transport investment. *The American Economic Review*, Vol. 59, Issue 2, pp. 251-260.

Vickrey, W. (1994). (edited by R. Arnott, A. B. Atkinson, K. Arrow, and J. H. Dreze). *Public Economics: Selected Papers by William Vickrey*. Cambridge University Press (www.uk.cambridge.org).

Victoria Transport Policy Institute (2003). Online TDM Encyclopedia. http://www.vtpi.org/tdm/.

Viti, F., S. F. Catalano, M. Li, C. Lindveld, and H. van Zuylen (2003). An optimization problem with dynamic route-departure time choice and pricing. Presented at the 82nd annual meeting of the Transportation Research Board, Washington, D.C.

von Stackelberg, H. (1934). Marktform und gleichgewicht. Vienna: Julius Springer. English edition: The theory of the market economy. Oxford University Press, Oxford, England, 1952.

Walters, A. A. (1961). The theory and measurement of private and social cost of highway congestion. *Econometrica,* Vol. 29, pp. 676-699.

Washington, S., J. D. Leonard II, C. A. Roberts, T. Young, D. Sperling, and J. Botha (1998). Forecasting Vehicle Modes of Operation Needed as Input to 'Modal' Emissions Models. *International Journal of Vehicle Design*, Vol. 20, Nos. 1-4 (Special Issue), pp. 351-359.

Wenzel T. and M. Ross (1996). Emissions from modern passenger cars with malfunctioning emissions controls. SAE paper 960067.

White, L. (1982). U.S. mobile source emissions regulation: the problems of implementation. *Journal of Policy Studies*, Vol. 11, pp. 77-85.

Wie, B., and R. L. Tobin (1998). Dynamic congestion pricing models for general traffic networks. *Transportation Research B,* Vol. 32, No. 5, pp. 313-327.

Williams, M. D., G. Thayer, M. J. Barth, and L. L. Smith (1999). The TRANSIMS approach to emissions estimation. Los Alamos National Laboratory.

Wolf, J., R. Guensler, S. Washington, W. Sarasua, C. Grant, S. Hallmark, M. Oliveira, M. Koutsak, R. Thittai, R. Funk, and J. Hsu (1999). Development of a comprehensive vehicle instrumentation package for monitoring individual tripmaking behavior. Georgia Institute of Technology, Final Report GTI-R - 99005.

Xu, Y. W., J. H. Wu, M. Florian, P. Marcotte, and D. L. Zhu (1999). Advances in the continuous dynamic network loading problem. *Transportation Science*, Vol. 33, No. 4, pp. 341-353.

Yang, H. (1997). Sensitivity analysis for the elastic-demand network equilibrium problem with applications. *Transportation Research B,* Vol. 31, No. 1, pp. 55-70.

Yang, H., and M. G. H. Bell (1997). Traffic restraint, road pricing, and network equilibrium. *Transportation Research B,* Vol. 31, No. 4, pp. 303-314.

Yang, H., and H. Huang (1998). Principle of marginal-cost pricing: how does it work in a general road network? *Transportation Research A*, Vol. 32, No. 1, pp. 45-54.

Yen, J. Y. (1970). An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, Vol. 27, No. 4, pp. 526-530.

Ziliaskopoulos, A. (1994). Optimum path algorithms on multidimensional networks: analysis, design, implementation and computational experience. Ph.D. Thesis, University of Texas at Austin.

Appendix A

Lemma 3.1: *The function* $e_n(i,t)$ *is a non-decreasing function of the number of augmentations. That is:* $e_{n-1}(i,t) \leq e_n(i,t)$.

Proof of Lemma 3.1

Assume without loss of generality that node-time pair (i,t) does not belong to the nth augmenting path P_n. We prove the lemma by contradiction. Assume that $e_{n-1}(i,t)$ decreases after the nth augmentation iteration, i.e. $e_{n-1}(i,t) > e_n(i,t)$ (by augmentation iteration, we refer to the process of computing a minimum cost augmenting path, augmenting flow on the path, and updating the residual network). Then (i,t) should be connected to at least one node-time pair $(j,u) \in P_n$ (see Figure A.1), for otherwise the nth augmentation iteration does not affect the minimum travel cost from (i,t) to q and $e_n(i,t) = e_{n-1}(i,t)$. Suppose that $e_n(i,t)$ is the length of a path P' from (i,t) to q which passes through at least one arc created after the nth augmentation iteration (this arc could be a reverse arc created after augmenting flow on P_n, or it could be a forward arc that restored capacity after augmenting flow on its corresponding reverse arc). Let P' be composed of three subpaths: a subpath of cost a connecting (i,t) to (j,u), a subpath consisting of arcs (created after the nth augmentation iteration) of total cost $-b$ and connecting (j,u) to another node-time pair (k,w) on P_n, and a subpath of cost c connecting (k,w) to the destination q. Thus, $e_n(i,t) = a - b + c$. Let d be the travel cost of the subpath from (j,u) to q along path P_n. We have:

(1) $e_{n-1}(i,t) \leq a + d$, since $e_{n-1}(i,t)$ is by definition the minimum travel cost from (i,t) to q before the nth augmentation iteration.

(2) $b+d \le c \Rightarrow d \le c-b$, for otherwise P_n would have used the subpath of length c to reach q.

From (1) and (2), $e_{n-1}(i,t) \le a-b+c$. Hence, $e_{n-1}(i,t) \le e_n(i,t)$. ■

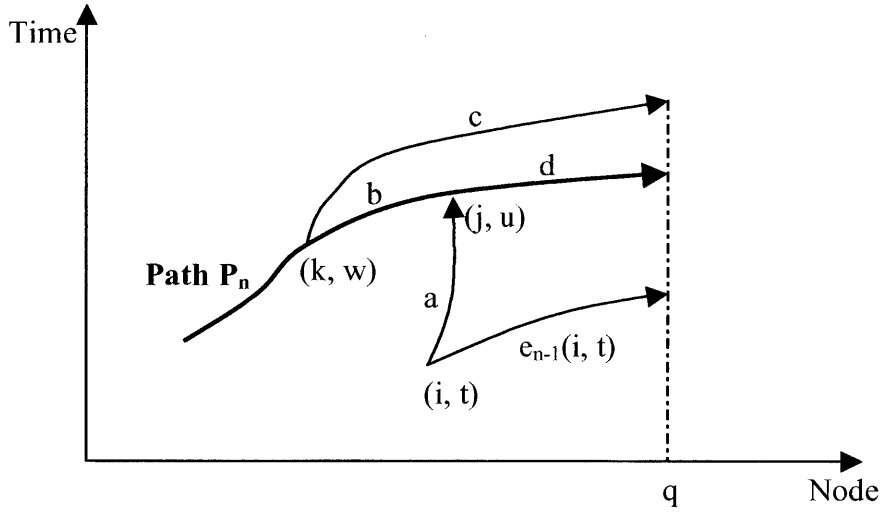


Figure A.1. Illustrative figure to prove the lower bound property.

Appendix B

Lemma 3.2: *Before Algorithm C terminates, there exists always a node-time pair* (i,t_i) *in the candidate set* C *such that: (1)* $\hat{\pi}(i,t_i)+\hat{e}(i,t_i)\le\pi^*$, *and (2)* (i,t_i) *is on a shortest path to* q.

Proof of Lemma 3.2

We prove the lemma by induction.

(1) Iteration 1: Initially, the candidate set C contains node-time pair $(s,0)$. Since $\hat{\pi}(s,0)+\hat{e}(s,0)=0+\hat{e}(s,0)\le\pi^*$ and $(s,0)$ is on a shortest path to q, the lemma is satisfied for the first iteration (i.e. first selection from the candidate set).

(2) Assume the lemma is satisfied up to iteration k. We show that the lemma is also satisfied at iteration $k+1$. Denote the node-time pair that satisfied this lemma at iteration k by (i,t_i). Therefore, at iteration k, we have $\hat{\pi}(i,t_i)+\hat{e}(i,t_i)\le\pi^*$ and (i,t_i) is on a shortest path to q. We distinguish two cases:

(i) If (i,t_i) were not selected from the candidate set C at iteration k, (i,t_i) would still be in C at iteration $k+1$. Since its cost label $\hat{\pi}(i,t_i)$ cannot increase, the lemma is still satisfied at iteration $k+1$.

(ii) If (i,t_i) were selected from the candidate set C at iteration k, (i,t_i) would try to update the cost labels of all node-time pairs in its forward star (and backward star, corresponding to reverse arcs with positive residual capacity). Since (i,t_i) belongs by assumption to a shortest path P to q, one of the nodes in the forward star (or backward star) of (i,t_i) is a node-

time pair (j,t_j) that belongs to P as well. If $\hat{\pi}(j,t_j)=\pi(j,t_j)$ before the update procedure from (i,t_i) (which means that there is another minimum cost path to q) and if (j,t_j) were selected before (i,t_i) is selected, then (j,t_j) would not be added to C after (i,t_i) is removed from C. However, when (j,t_j) was selected from C, one of the nodes in the forward star (or backward star) of (j,t_j) is a node-time pair (l,t_l) that belongs to P as well. If $\hat{\pi}(l,t_l)=\pi(l,t_l)$ before the update procedure from (j,t_j) and if (l,t_l) were selected before (j,t_j) is selected, then (l,t_l) would not be added to C after (j,t_j) is removed from C. Since there is a finite number of node-time pairs on P after (i,t_i), one could apply this argument consecutively for those node-time pairs after (i,t_i) on P until one eventually reaches a node-time pair (p,t_p) whose cost label $\hat{\pi}(p,t_p)$ is or gets updated to $\pi(p,t_p)$ and (p,t_p) is in C when (i,t_i) is selected from C. This must be true for otherwise this implies that the destination node q was selected (since the path P ends at q) and the algorithm would have terminated. Therefore, at iteration $k+1$ there is a node-time pair (p,t_p) such that $\hat{\pi}(p,t_p)+\hat{e}(p,t_p)\leq\pi^*$ and (p,t_p) is on a shortest path to q, and the lemma is still satisfied at iteration $k+1$.

■

Corollary 3.3: *Every node-time pair* (j,t_j) *selected from* C *is such that:* $\hat{\pi}(j,t_j)+\hat{e}(j,t_j)\leq\pi^*$.

Proof of Corollary 3.3

When a node-time pair (j,t_j) is selected from C, there exists a node-time pair (i,t_i) in C such that: $\hat{\pi}(i,t_i)+\hat{e}(i,t_i)\le\pi^*$. Since Algorithm C selects nodes by increasing order of $\hat{\lambda}$, $\hat{\lambda}(j,t_j)=\hat{\pi}(j,t_j)+\hat{e}(j,t_j)\le\hat{\lambda}(i,t_i)=\hat{\pi}(i,t_i)+\hat{e}(i,t_i)\le\pi^*$. ■

Appendix C

Lemma 3.4: *For any augmenting path, the arrival time at the destination is greater than the arrival time at any intermediate node-time pair on the augmenting path.*

Proof of Lemma 3.4

We prove the lemma by induction.

(1) First augmenting path: Since all link travel times are initially positive before augmenting any flow, the first augmenting path consists of links with positive travel times. Hence, $\pi(q) > \pi(i,t_i) = t_i$, where (i,t_i) is any intermediate node-time pair on the first augmenting path.

(2) k^{th} augmenting path: Assume the lemma is satisfied up to the k^{th} augmentation. We prove that the lemma is satisfied for the $(k+1)^{st}$ augmenting path.

By Lemma 3.1, the arrival time at the destination is a non-decreasing function of the number of augmentation iterations. Thus, $\pi_{k+1}(q) \geq \pi_k(q) \geq \ldots \geq \pi_1(q)$, where $\pi_n(q)$ denotes the arrival time at the destination for the n^{th} augmenting path. By the induction hypothesis, for $n = 1,\ldots,k$, $\pi_n(q)$ is greater than the arrival time labels $\pi(i,t_i) = t_i$ of all intermediate node-time pairs (i,t_i) on the n^{th} augmenting path. Thus, for $n = 1,\ldots,k$, $\pi_n(q)$ is also greater than the arrival time labels $\pi(i,t_i) = t_i$ of all intermediate node-time pairs (i,t_i) on the first n augmenting paths. Since $\pi_{k+1}(q) \geq \pi_k(q)$, $\pi_{k+1}(q)$ is greater than the arrival time labels $\pi(i,t_i) = t_i$ of all intermediate node-time pairs (i,t_i) on the first k augmenting paths. This means that at the beginning of the $(k+1)^{st}$ augmentation, all reverse arcs in the residual network emanate from node-time pairs (j,t_j) such that $t_j < \pi_{k+1}(q)$. Therefore, all node-time pairs (i,t_i) belonging to the $(k+1)^{st}$ augmenting path are such that $\pi(i,t_i) = t_i < \pi_{k+1}(q)$, and the lemma is proved for the $(k+1)^{st}$ augmenting path. ■

Appendix D

In this appendix, we discuss in some detail the theory of marginal cost pricing. The marginal social cost of traveling due to an additional user is equal to the sum of an internal cost (the average cost perceived by the user, which is also called the marginal private cost) and an external cost (the delay costs incurred by all other users due to an additional trip). The theory of marginal cost pricing, dating back to Pigou (1920) and further explored by Walters (1961) and Vickrey (1967), postulates that in order to maximize the economic efficiency of trip-making, a toll equal to the difference between the marginal social cost and marginal private cost at the optimal volume of traffic should be levied. Let TC, MSC, and AC denote the total cost, marginal social cost, and average (private) cost, and let V denote the volume of traffic. Then:

$$TC = V \times AC$$

$$MSC = \underbrace{\frac{d(TC)}{dV}}_{\text{Marginal social cost}} = \underbrace{AC}_{\text{Internal cost}} + \underbrace{V \times \frac{d(AC)}{dV}}_{\text{External cost}}$$

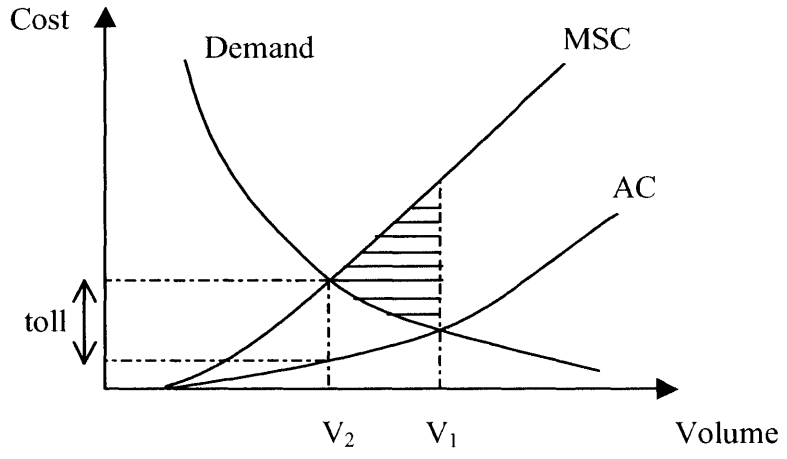


Figure D.1. Marginal cost pricing.

The no-toll situation corresponds to an equilibrium volume V_1, at which average cost is equated to marginal benefit as given by the height of the demand function. It can be seen that at V_1 the marginal social cost exceeds the marginal benefit, thus resulting in economic inefficiencies. At volume V_2, marginal benefit is equal to marginal social cost, and thus a dead-weight loss equal to the shaded area can be eliminated by moving from V_1 to V_2. The optimal toll that can establish equilibrium volume V_2 is thus equal to the difference between marginal social cost and average cost at traffic volume V_2. The reader is referred to Yang and Huang (1998) for more information about the theory of marginal cost pricing as applied to a general road network.

Marginal cost pricing is a first-best solution to congestion provided the following assumptions are satisfied (Emmerink (1998)): rational individual behavior, full information on all costs involved, applicability of prices to all network links, technical feasibility of pricing, and low transaction costs.

Appendix E

In this appendix, we present the derivations of the expression $\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')}$ that was used to derive the gradient expression in Section 5.2.3.1 (for route choice only) and the expression $\frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}$ that was used to derive the gradient expression in Section 5.2.3.2 (for route and departure time choices) for the joint logit and two nested logit models of route and departure time choices.

E.1 Derivation of the Term $\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')}$

To derive the term $\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')}$, we first derive its inverse $\frac{\partial h^{r's'*}(p',t')}{\partial d^{rs}(p,t)}$, which can be expressed as follows:

$$\frac{\partial h^{r's'*}(p',t')}{\partial d^{rs}(p,t)} = \frac{\partial\left(\sum_{m'} h_{m'}^{r's'*}(p',t')\right)}{\partial d^{rs}(p,t)} = \sum_{m'} \frac{\partial h_{m'}^{r's'*}(p',t')}{\partial d^{rs}(p,t)}. \tag{E.1}$$

The term $\frac{\partial h_{m'}^{r's'*}(p',t')}{\partial d^{rs}(p,t)}$ can be expressed as follows:

$$\frac{\partial h_{m'}^{r's'*}(p',t')}{\partial d^{rs}(p,t)} = \frac{\partial\left(D_{m'}^{r's'}(t')P_{m'}^{r's'}(p',t')\right)}{\partial d^{rs}(p,t)} = D_{m'}^{r's'}(t')\frac{\partial P_{m'}^{r's'}(p',t')}{\partial d^{rs}(p,t)}, \tag{E.2}$$

where the demand $D_{m'}^{r's'}(t')$ is fixed, and for a C-logit route choice model $P_{m'}^{r's'}(p',t')$ is given by:

$$P_{m'}^{r's'}(p',t') = \frac{\exp\left(V_{m'}^{r's'}(p',t') - CF^{r's'}(p')\right)}{\sum_{y \in R^{r's'}} \exp\left(V_{m'}^{r's'}(y,t') - CF^{r's'}(y)\right)}. \tag{E.3}$$

Therefore, $\frac{\partial P_{m'}^{r's'}(p',t')}{\partial d^{rs}(p,t)}$ can be expressed as follows:

$$\frac{\partial P_{m'}^{r's'}(p',t')}{\partial d^{rs}(p,t)} = \sum_{y \in R^{r's'}} \frac{\partial P_{m'}^{r's'}(p',t')}{\partial V_{m'}^{r's'}(y,t')} * \frac{\partial V_{m'}^{r's'}(y,t')}{\partial d^{rs}(p,t)}. \tag{E.4}$$

We have:

$$\frac{\partial V_{m'}^{r's'}(y,t')}{\partial d^{rs}(p,t)} = \begin{cases} \alpha_{m'd}, & \text{if } y = p, t' = t, (r',s') = (r,s) \\ 0, & \text{otherwise.} \end{cases} \tag{E.5}$$

Substituting expression (E.5) in (E.4), we obtain:

$$\frac{\partial P_{m'}^{r's'}(p',t')}{\partial d^{rs}(p,t)} = \begin{cases} \frac{\partial P_{m'}^{rs}(p',t)}{\partial V_{m'}^{rs}(p,t)} * \alpha_{m'd}, & \text{if } t' = t, (\mathrm{r}',\mathrm{s}') = (r,s) \\ 0, & \text{otherwise.} \end{cases} \tag{E.6}$$

Substituting expression (E.6) in (E.2), we obtain:

$$\frac{\partial h_{m'}^{r's'*}(p',t')}{\partial d^{rs}(p,t)} = \begin{cases} D_{m'}^{rs}(t) * \frac{\partial P_{m'}^{rs}(p',t)}{\partial V_{m'}^{rs}(p,t)} * \alpha_{m'd}, & \text{if } t' = t, (\mathrm{r}',\mathrm{s}') = (r,s) \\ 0, & \text{otherwise.} \end{cases} \tag{E.7}$$

Substituting expression (E.7) in (E.1), $\frac{\partial h^{r's'*}(p',t')}{\partial d^{rs}(p,t)}$ can then be expressed as follows:

$$\frac{\partial h^{r's'*}(p',t')}{\partial d^{rs}(p,t)} = \begin{cases} \sum_{m'} \left(D_{m'}^{rs}(t) * \frac{\partial P_{m'}^{rs}(p',t)}{\partial V_{m'}^{rs}(p,t)} * \alpha_{m'd} \right), & \text{if } t' = t, (\mathrm{r}',\mathrm{s}') = (r,s) \\ 0, & \text{otherwise.} \end{cases} \tag{E.8}$$

Finally, $\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')}$ is given by:

$$\frac{\partial d^{rs}(p,t)}{\partial h^{r's'*}(p',t')} = \begin{cases} \dfrac{1}{\sum_{m'} \alpha_{m'd} D_{m'}^{rs}(t) \dfrac{\partial P_{m'}^{rs}(p',t)}{\partial V_{m'}^{rs}(p,t)}}, & \text{if } t' = t, (r',s') = (r,s) \\ \text{undefined}, & \text{otherwise.} \end{cases} \tag{E.9}$$

E.2 Derivation of the Term $\frac{\partial P_m^{rs}(p,t)}{\partial \tilde{V}_m^{rs}(y,k)}$

For simplicity, in what follows we drop the indices referring to user class and O-D pair.

E.2.1 Joint Logit Model

For a joint logit model,

$$P(p,t) = \frac{\exp\left(\tilde{V}(p) + \tilde{V}(t) + \tilde{V}(p,t)\right)}{\sum_{(p',t')} \exp\left(\tilde{V}(p') + \tilde{V}(t') + \tilde{V}(p',t')\right)}. \tag{E.10}$$

We consider two cases separately:

Case 1: $(p,t) = (y,k)$

$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)}$ is then given by the following expression:

$$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)} = \frac{\partial P(p,t)}{\partial \tilde{V}(p,t)} = \frac{\left(\exp\left(\tilde{V}(p) + \tilde{V}(t) + \tilde{V}(p,t)\right) \sum_{(p',t')} \exp\left(\tilde{V}(p') + \tilde{V}(t') + \tilde{V}(p',t')\right)\right)}{\left[\sum_{(p',t')} \exp\left(\tilde{V}(p') + \tilde{V}(t') + \tilde{V}(p',t')\right)\right]^2} - \left[\exp\left(\tilde{V}(p) + \tilde{V}(t) + \tilde{V}(p,t)\right)\right]^2$$

$$= P(p,t) - [P(p,t)]^2 = P(p,t)[1 - P(p,t)]. \tag{E.11}$$

Case 2: $(p,t) \neq (y,k)$

$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)}$ is then given by the following expression:

$$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)} = \frac{-\exp\left(\tilde{V}(p)+\tilde{V}(t)+\tilde{V}(p,t)\right)*\exp\left(\tilde{V}(y)+\tilde{V}(k)+\tilde{V}(y,k)\right)}{\left[\sum_{(p',t')}\exp\left(\tilde{V}(p')+\tilde{V}(t')+\tilde{V}(p',t')\right)\right]^2} = -P(p,t)P(y,k). \tag{E.12}$$

In summary, $\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)}$ is given by the following expression:

$$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)} = \begin{cases} P(p,t)[1-P(p,t)] & \text{if } (p,t)=(y,k) \\ -P(p,t)P(y,k) & \text{if } (p,t)\neq(y,k). \end{cases} \tag{E.13}$$

E.2.2 Nested Logit Model 1

For nested logit model 1, the joint probability $P(p,t)$ can be derived as follows:

$$\begin{aligned} P(p,t) = P(p/t)*P(t) &= \frac{\exp\left(\mu_p\left(\tilde{V}(p,t)+\tilde{V}(p)\right)\right)}{\sum_{p'}\exp\left(\mu_p\left(\tilde{V}(p',t)+\tilde{V}(p')\right)\right)} * \frac{\exp\left(\mu_t\left(\tilde{V}(t)+V'(t)\right)\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+V'(t')\right)\right)} \\ &= \frac{\exp\left(\mu_p\left(\tilde{V}(p,t)+\tilde{V}(p)\right)\right)}{\exp\left(\mu_p V'(t)\right)} * \frac{\exp\left(\mu_t\left(\tilde{V}(t)+V'(t)\right)\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+V'(t')\right)\right)} \\ &= \frac{\exp\left(\mu_p\left(\tilde{V}(p,t)+\tilde{V}(p)\right)+\mu_t\tilde{V}(t)+(\mu_t-\mu_p)V'(t)\right)}{\sum_{t'}\exp\left(\mu_t\left(\tilde{V}(t')+V'(t')\right)\right)}. \end{aligned} \tag{E.14}$$

We consider four cases separately:

Case 1: $t = k$ and $p = y$

In this case, $\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)}$ can be expressed as follows:

$$\frac{\partial P(p,t)}{\partial \widetilde{V}(y,k)} = \frac{\partial P(p,t)}{\partial \widetilde{V}(p,t)} = \frac{\left(\begin{array}{l} \left(\mu_p + (\mu_t - \mu_p)\dfrac{\partial V'(t)}{\partial \widetilde{V}(p,t)}\right)\exp\left(\begin{array}{l}\mu_p\left(\widetilde{V}(p,t)+\widetilde{V}(p)\right)+\mu_t\widetilde{V}(t)\\ +(\mu_t-\mu_p)V'(t)\end{array}\right)* \\ \sum_{t'}\exp\left(\mu_t\left(\widetilde{V}(t')+V'(t')\right)\right) - \exp\left(\begin{array}{l}\mu_p\left(\widetilde{V}(p,t)+\widetilde{V}(p)\right)+\mu_t\widetilde{V}(t)\\ +(\mu_t-\mu_p)V'(t)\end{array}\right)* \\ \sum_{t'}\left(\mu_t * \dfrac{\partial V'(t')}{\partial \widetilde{V}(p,t)}\exp\left(\mu_t\left(\widetilde{V}(t')+V'(t')\right)\right)\right) \end{array}\right)}{\left[\sum_{t'}\exp\left(\mu_t\left(\widetilde{V}(t')+V'(t')\right)\right)\right]^2}. \tag{E.15}$$

But $V'(t) = \frac{1}{\mu_p}\ln\sum_{p'}\exp\left(\mu_p\left(\widetilde{V}(p')+\widetilde{V}(p',t)\right)\right)$. Therefore, $\frac{\partial V'(t)}{\partial \widetilde{V}(y,k)}$ can be expressed as follows:

$$\frac{\partial V'(t)}{\partial \widetilde{V}(y,k)} = \begin{cases} \dfrac{1}{\mu_p}\dfrac{\mu_p\exp\left(\mu_p\left(\widetilde{V}(y)+\widetilde{V}(y,t)\right)\right)}{\sum_{p'}\exp\left(\mu_p\left(\widetilde{V}(p')+\widetilde{V}(p',t)\right)\right)} = P(y/t), & \text{if } t=k \\ 0, & \text{otherwise} \end{cases} \tag{E.16}$$

Therefore, we have:

$$\begin{aligned} \frac{\partial P(p,t)}{\partial \widetilde{V}(y,k)} &= \frac{\partial P(p,t)}{\partial \widetilde{V}(p,t)} \\ &= \left(\mu_p + (\mu_t-\mu_p)P(p/t)\right)P(p,t) - P(p,t)*\frac{\mu_t P(p/t)\exp\left(\mu_t\left(\widetilde{V}(t)+V'(t)\right)\right)}{\sum_{t'}\exp\left(\mu_t\left(\widetilde{V}(t')+V'(t')\right)\right)} \\ &= \left(\mu_p + (\mu_t-\mu_p)P(p/t)\right)P(p,t) - P(p,t)*\mu_t P(p/t)P(t) \\ &= \left[\mu_p + (\mu_t-\mu_p)P(p/t) - \mu_t P(p,t)\right]P(p,t). \end{aligned} \tag{E.17}$$

Case 2: $t = k$ and $p \neq y$

In this case, $\frac{\partial P(p,t)}{\partial \widetilde{V}(y,k)}$ is given by the following expression:

$$\frac{\partial P(p,t)}{\partial \widetilde{V}(y,k)} = \frac{\partial P(p,t)}{\partial \widetilde{V}(y,t)} = \frac{\left(\begin{array}{l} \left((\mu_t - \mu_p) \dfrac{\partial V'(t)}{\partial \widetilde{V}(y,t)} \right) \exp\left(\begin{array}{l} \mu_p \left(\widetilde{V}(p,t) + \widetilde{V}(p) \right) + \mu_t \widetilde{V}(t) \\ + (\mu_t - \mu_p) V'(t) \end{array} \right) * \\ \sum_{t'} \exp\left(\mu_t \left(\widetilde{V}(t') + V'(t') \right) \right) - \exp\left(\begin{array}{l} \mu_p \left(\widetilde{V}(p,t) + \widetilde{V}(p) \right) + \mu_t \widetilde{V}(t) \\ + (\mu_t - \mu_p) V'(t) \end{array} \right) * \\ \mu_t * \dfrac{\partial V'(t)}{\partial \widetilde{V}(y,t)} \exp\left(\mu_t \left(\widetilde{V}(t) + V'(t) \right) \right) \end{array} \right)}{\left[\sum_{t'} \exp\left(\mu_t \left(\widetilde{V}(t') + V'(t') \right) \right) \right]^2}$$

$$= (\mu_t - \mu_p) P(y/t) P(p,t) - P(p,t) * \mu_t P(y/t) P(t)$$

$$= \left[\mu_t - \mu_p - \mu_t P(t) \right] P(y/t) P(p,t). \qquad \text{(E.18)}$$

Case 3: $t \neq k$ and $p = y$

In this case, $\dfrac{\partial P(p,t)}{\partial \widetilde{V}(y,k)}$ is given by the following expression:

$$\frac{\partial P(p,t)}{\partial \widetilde{V}(y,k)} = \frac{\partial P(p,t)}{\partial \widetilde{V}(p,k)} = \frac{\left(\begin{array}{l} -\exp\left(\mu_p \left(\widetilde{V}(p,t) + \widetilde{V}(p) \right) + \mu_t \widetilde{V}(t) + (\mu_t - \mu_p) V'(t) \right) * \\ \mu_t * \dfrac{\partial V'(k)}{\partial \widetilde{V}(p,k)} \exp\left(\mu_t \left(\widetilde{V}(k) + V'(k) \right) \right) \end{array} \right)}{\left[\sum_{t'} \exp\left(\mu_t \left(\widetilde{V}(t') + V'(t') \right) \right) \right]^2} \qquad \text{(E.19)}$$

$$= -P(p,t) * \mu_t P(p/k) P(k) = -\mu_t P(p,k) P(p,t).$$

Case 4: $t \neq k$ and $p \neq y$

In this case, $\dfrac{\partial P(p,t)}{\partial \widetilde{V}(y,k)}$ is given by the following expression:

$$\frac{\partial P(p,t)}{\partial \widetilde{V}(y,k)} = \frac{\left(\begin{array}{l} -\exp\left(\mu_p \left(\widetilde{V}(p,t) + \widetilde{V}(p) \right) + \mu_t \widetilde{V}(t) + (\mu_t - \mu_p) V'(t) \right) * \\ \mu_t * \dfrac{\partial V'(k)}{\partial \widetilde{V}(y,k)} \exp\left(\mu_t \left(\widetilde{V}(k) + V'(k) \right) \right) \end{array} \right)}{\left[\sum_{t'} \exp\left(\mu_t \left(\widetilde{V}(t') + V'(t') \right) \right) \right]^2} \qquad \text{(E.20)}$$

$$= -P(p,t) * \mu_t P(y/k) P(k) = -\mu_t P(y,k) P(p,t).$$

In summary, $\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)}$ is given by the following expression:

$$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)} = \begin{cases} \left(\mu_p + (\mu_t - \mu_p)P(p/t) - \mu_t P(p,t)\right)P(p,t) & \text{if } p = y, t = k \\ \left(\mu_t - \mu_p - \mu_t P(t)\right)P(y/t)P(p,t) & \text{if } p \neq y, t = k \\ -\mu_t P(p,k)P(p,t) & \text{if } p = y, t \neq k \\ -\mu_t P(y,k)P(p,t) & \text{if } p \neq y, t \neq k. \end{cases} \tag{E.21}$$

E.2.3 Nested Logit Model 2

For nested logit model 2, the joint probability $P(p,t)$ can be expressed as follows:

$$P(p,t) = \frac{\exp\left(\mu_t\left(\tilde{V}(p,t) + \tilde{V}(t)\right) + \mu_p \tilde{V}(p) + (\mu_p - \mu_t)V'(p)\right)}{\sum_{p'} \exp\left(\mu_p\left(\tilde{V}(p') + V'(p')\right)\right)}. \tag{E.22}$$

By symmetry to the derivations in nested logit model 1, we have:

$$\frac{\partial P(p,t)}{\partial \tilde{V}(y,k)} = \begin{cases} \left(\mu_t + (\mu_p - \mu_t)P(t/p) - \mu_p P(p,t)\right)P(p,t) & \text{if } p = y, t = k \\ -\mu_p P(y,t)P(p,t) & \text{if } p \neq y, t = k \\ \left(\mu_p - \mu_t - \mu_p P(p)\right)P(k/p)P(p,t) & \text{if } p = y, t \neq k \\ -\mu_p P(y,k)P(p,t) & \text{if } p \neq y, t \neq k. \end{cases} \tag{E.23}$$