The Dissertation Committee for Venktesh Pandey
Certifies that this is the approved version of the following dissertation:

Dynamic Pricing and Long-term Planning Models for Managed Lanes with Multiple Entrances and Exits

APPROVED BY
SUPERVISING COMMITTEE:

Stephen D. Boyles (Supervisor)

Chandra Bhat

Christian Claudel

Peter Stone

John Hasenbein

Dynamic Pricing and Long-term Planning Models for Managed Lanes with Multiple Entrances and Exits

by

Venktesh Pandey

Dissertation

Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

The University of Texas at Austin
May 2020

To my family

Acknowledgements

This dissertation has been influenced by several people who deserve special mention. First, I am very grateful to my advisor, Steve Boyles, for all his support, mentorship, and guidance throughout the process of this dissertation. He showed profound belief in my work and my abilities and provided a great environment to improve my research skills as a graduate student. I would not be where I am at if it was not for his support. Thank you, Steve. I would also like to extend my sincere thanks to my dissertation committee members—Chandra Bhat, Christian Claudel, Peter Stone, and John Hasenbein—for all their advice and feedback and for the invaluable insight through courses that started the process of this dissertation.

I gratefully acknowledge the financial support from the National Science Foundation, Texas Department of Transportation, and Data-Supported Transportation Operations and Planning University Transportation Center. I am also thankful to the staff at the Network Modeling Center, especially Natalia Ruiz Juri and Ken Perrine, for providing me the computational resources and data to work on various projects. I also very much appreciate the role of administrative assistants, especially Lisa Macias, Vicki Simpson, and Caitlan McCollum, who helped me go through all administrative hurdles.

I am also thankful to other collaborators and mentors who helped me through my graduate school. Special thanks to Tarun for providing guidance on several technical topics and offering a methodological perspective on the last chapter of this dissertation. I am also thankful to Michael and Ehsan for their colleagueship and collaboration during our Ph.D. process, and to Julien and Andrea at IBM Research for giving me the opportunity to work with real-world datasets. I am also thankful to Maura Borrego for her guidance in improving my teaching skills and to Ana Dison for giving me the opportunity to be engaged with the Graduates Linked with Undergraduates in Engineering mentoring program over different semesters.

The completion of my dissertation would not have been possible without the support of my friends and colleagues in the transportation program at UT. Special thanks to Rydell for all the fun times planning for Traffic Bowl meetings, the spontaneous Sunday runs, and

random chats in our office; Rachel for ensuring that we stayed sane through the Ph.D. process; Kristie, Alice, Felipe, Patricia, Vivek, Murthy, Ramez, Natalia, Pavle, and Michele for being excellent colleagues to work with and share a laugh; and the present and past members of the SPARTA lab—John, Sudesh, Prashanth, Cesar, Priyadarshan, Can, Patrick, Will, Carlin, Rahul, Tengkuo, Rishabh, and Karthik—for their camaraderie and all the fun times chatting about technical (and non-technical) topics at group meetings and tea times. You all made going through graduate school a much memorable experience. Thank you!

I am also grateful to my friends outside of the world of transportation who helped me keep going and for all the beautiful memories. Special thanks to Vatsal, Devesh, Priyanka, Nitin, Pranav, Bharath, Esha, Eddy, Akanksha, Preeti, Parshu, Jeremy, and Jon for everything ranging from spontaneous hangouts and game nights to being there for each other through our difficult times. I am also incredibly thankful to Luca for his amazing friendship and support over the last few years—could not have done it without you. I am also grateful to Sheilah, Jaya, Paolo, Rebecca, Luisa, Kyle, and Elena for making me have a great family experience while away from home. Many thanks also to my running group, Run for India, for keeping me motivated for running.

I am also thankful for my undergraduate mentees—Evana, Andres, Abby, Anne, Jordan, and Manisha—who inspired me with their curiosity to learn. I am grateful to all the students I had the opportunity to teach and mentor–you all inspire me and reinforce how much I enjoy the process of teaching and learning together. I also extend my warm thanks to the staff at The Flightpath Coffeehouse where I spent a large chunk of my time drafting this dissertation.

Last and the most important, I am deeply indebted to my family back in India for all their sacrifices so I can have a better education, for showing me their love and support every day, and for reminding me more than often that I have got this. I dedicate this dissertation to them.

I am very grateful to have each and every one of you in my life. Thank you for making this possible.

Abstract

Dynamic Pricing and Long-term Planning Models for Managed Lanes with Multiple Entrances and Exits

Venktesh Pandey, Ph.D.

The University of Texas at Austin, 2020

Supervisor: Stephen D. Boyles

Express lanes or priced managed lanes provide a reliable alternative to travelers by charging dynamic tolls in exchange for traveling on lanes with no congestion. These lanes have various locations of entrances and exits and allow travelers to adapt their route based on the toll and travel time information received at a toll gantry. In this dissertation, we incorporate this adaptive lane choice behavior in improving the dynamic pricing and long-term planning models for managed lanes with multiple entrances and exits.

Lane choice of travelers minimizing their disutility is affected by the real-time information about tolls and travel time through variable message signs and perceived information from past experiences. In this dissertation, we compare various adaptive lane choice models differing in their reliance on real-time information or historic information or both. We propose a *decision route* lane choice model that efficiently compares the disutility over multiple routes on an express lane. Assuming drivers disutility is only affected by tolls and travel times, we show that the decision route model generates only up to 0.93% error in expected costs compared to the optimal adaptive lane choice model, making it a suitable choice for modeling lane choice of travelers.

Next, using the decision route lane choice framework, we improve the current dynamic pricing models for express lanes that commonly ignore adaptive lane choice, assume simplified traffic dynamics, and/or are based on simplified heuristics. Formulating the dynamic pricing problem as an MDP, we optimize the tolls for various objectives including maximizing revenue and minimizing total system travel time (TSTT). Three solution algorithms

are evaluated: (a) an algorithm based on value-function approximation, (b) a multiagent reinforcement learning algorithm with decentralized tolling at each gantry, and (c) a deep reinforcement learning assuming partial observability of traffic state. These algorithms are shown to outperform other heuristics such as feedback control heuristics by generating up to 10% higher revenues and up to 9% lower delays. Our findings also reveal that the revenue-maximizing optimal policies follow a "jam-and-harvest" behavior where the toll-free lanes are pushed towards congestion in the earlier time steps to generate higher revenue later, a characteristic not observed for the policies minimizing TSTT. We use reward shaping methods to overcome the undesired behavior of toll policies and confirm transferability of the algorithms to new input domains. We also offer recommendations on real-time implementations of pricing algorithms based on solving MDPs.

Last, we incorporate adaptive lane choice in existing long-term planning models for express lanes which commonly represent these lanes as fixed-toll facilities and ignore *en route* adaptation of lane choices. Defining the improved model as an equilibrium over adaptive lane choices of self-optimizing travelers and formulating it as a convex program, we show that long-term traffic forecasts can be underestimated by up to 45% if adaptive route choice is ignored. For solving the equilibrium, we develop a gradient-projection algorithm which is shown to be efficient than existing link-state algorithms in the literature. Additionally, we estimate the sensitivity of equilibrium expected costs with demand variation by formulating it as a convex program solved using a variant of the gradient projection algorithm proposed earlier. This analysis simplifies a complex express lane network as a single directed link, allowing integration of adaptive lane choice for planning of express lanes without significantly altering the components of traditional planning models.

Overall these models improve the state-of-the-art of pricing and planning for managed lanes useful for evaluating future express lane projects and for operations of express lanes with multiple objectives.

Keywords: Managed lanes, Dynamic pricing, Markov decision process, Deep reinforcement learning, User equilibrium with recourse, Network contraction

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Managing traffic congestion is a growing challenge for transportation planners, traffic operators, and engineers. The United States lost around $121 billion in net worth in the year 2011 directly attributable to congestion on roadway facilities [2]. A recent trend in managing freeway congestion is to prioritize improving travel time reliability while trying to minimize user delay in the network [3]. Travelers using transportation networks want to reach their destination on time and a reliable freeway corridor guarantees that their experienced delay does not exceed a threshold.

A trending way to improve travel time reliability is by constructing new managed lanes or repurposing existing lanes as managed lanes on freeway. A managed lane (ML) project sets apart a set of lanes "where operational strategies are proactively implemented and managed in response to changing conditions" to provide reliable travel time to the road user [4]. A subcategory of managed lanes are priced managed lanes, which are also referred as express lanes or high-occupancy/toll (HOT) lanes, where the user has to pay a toll for utilizing the facility.

Priced managed lanes are increasingly being used by many cities around the United States. As of January 2019, there are 41 active managed lane projects across the United States [5]. These include North Tarrant Express and LBJ TEXpress corridor in Dallas Fort-Worth, I-85 lanes in Atlanta, I-495 lanes in Northern Virginia, I-15 lanes in San Diego, MoPaC Expressway in Austin, and Katy Freeway in Houston. These lanes exploit users' willingness to pay for saving travel time and charge toll rates which may vary with the time-of-day or dynamically based on the congestion pattern. These lanes also generate revenue for infrastructure projects and promote the usage of transit by providing faster travel time for transit vehicles.

As managed lanes solve congestion problems and gain popularity, its infrastructure has also become complex. Managed lanes on a corridor now have multiple assess locations, and can span an entire corridor across a city. The LBJ TEXpress Lanes, a corridor of managed lanes in Dallas, Texas, feature 15 entrance ramps and 16 exit ramps along the 13.3-mile stretch of the roadway [6]. Networks of managed lanes can exist, with one corridor merging into another. Given the widespread adoption of electronic tolling and dynamic tolling based on real-time measurements, tolls on managed lanes can now adapt to current traffic conditions.

Furthermore, the deployment of extensive sensor networks and penetration of location-based services such as global positioning systems (GPS) on mobile phones have enabled conveying real-time information to the travelers. In the future, we also expect connected vehicles to obtain real-time updates about travel time and tolls using vehicle-to-vehicle or vehicle-to-infrastructure connections. The provision of real-time information allows travelers to adapt their routes based on the information received, which makes predicting the number of travelers using the managed lanes uncertain complicating the operations and planning of these lanes.

Additionally, public private partnerships (PPP) are commonly being used to finance the construction of these lanes. Under PPP, a private entity handles the design, construction, planning, operations, and management of managed lanes over a time period typically spanning multiple decades. Different agencies assign different priorities to various objectives for managed lane operations. For example, a private entity might prioritize revenue generation over ensuring least possible delay for travelers while doing toll operations. Managed lane operations have thus become more complicated than in the past as they are now multiobjective with varying priorities for different objectives including enhancing the HOT lane efficiency and utilization, providing travel time reliability, reducing total delay, and yielding sufficient revenue to offset the lifecycle costs of the project [7]. In addition, there are emerging social equity concerns with the usage of managed lanes: is the social benefit of reduced delay equally distributed across travelers from all social classes?[1].

[1]While express lanes are nicknamed "Lexus" lanes mocking the typical affluence of its users, there is conflicting evidence on the income levels of the users of these lanes [8, 9]

1.2 Motivation

This dissertation is motivated by three key challenges which will be the focus in the remainder of this document.

First, from the perspective of planning and operations, understanding how travelers make lane choice decisions under the presence of real-time information is critical for the success of managed lanes. For managed lanes with multiple entrance and exit locations, the driver lane choice behavior can be complex. Travelers decide where and when they enter and exit the express lanes given the current information on toll prices and travel time savings. Current lane-choice models make simplified assumptions on driver behavior, like a traveler only compares two choices at each diverge, a traveler does not exit the managed lane once they enter the lane until their exit is reached, and/or travel time on the general purpose lanes (GPL) is unaffected by the shift of travelers to the express lanes [10, 11, 12].

Inaccurate predictions of lane choice decisions of travelers can impact the traffic and revenue forecasts and also the investment decisions for an express lane. For example, according to a recent article in The Seattle Times, the I-405 express toll lanes in Seattle are operating below the desired speed limit 90% of the time [13]. In another recent instance, the toll rates on the 66 Express lanes in Virginia were raised to $47.5 during peak hours to maintain free flow on the express lanes [14]. These instances demonstrate the need for better driver behavior model which can predict lane choices for travelers accurately.

Second, building on accurate driver behavior models, there is a need for pricing models which can optimize toll prices for a certain objective. Most of the current dynamic pricing strategies utilize real-time measurements for dynamic pricing and are heuristic in nature: the decision to increase or decrease the toll is often made using a pre-determined threshold. An example of a heuristic strategy based on density measurements made using a loop detector data is shown in Figure 1.1, where the choice of whether or not to increase the toll is based on the measurements of current density of the roadway.

These heuristics can be potentially improved and tolls can be dynamically updated to achieve a particular objective. Models for managed lanes with a single entrance and a single exit have been extensively studied. Such systems are easier to model because there is

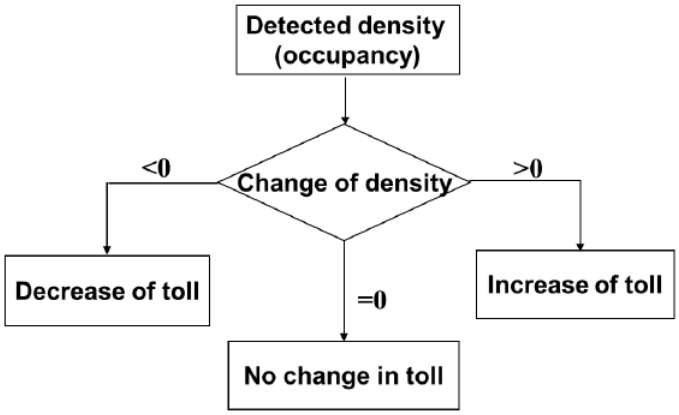


Figure 1.1: Commonly used heuristic for dynamic tolling of managed lanes (Source- Michalaka et al. [1])

only one decision point for the traveler and the tolls influence traveler's decision only once. However, for managed lanes with multiple entrances and multiple exits, there are multiple decision points located at each diverge location. In such cases, it is complicated to model the behavior of a traveler and update the tolls that still achieve a particular system-wide objective. Furthermore, the mechanism to handle the tradeoff between multiple objectives for dynamic pricing has not been studied.

Last, current models used for long-term planning of managed lanes model the lanes as toll facilities with static tolls and ignore the *en route* changes to route choice made by the availability of real-time information. In the current literature, equilibrium models and algorithms have been proposed that consider adaptive route choice under the presence of network uncertainties [15, 16, 17]. However, these models need to be adapted for cases where users of the managed lane corridor belong to different travel classes. Additionally, there is a need for integrating the equilibrium model with adaptive route choice into the traditional planning models and software.

Given these complications, how should a planning agency handle day-to-day operations and long-term planning for managed lanes of the future? In this dissertation, we focus on improving the dynamic pricing and long-term planning models by considering *en route* changes to the route choice of the travelers when subject to real-time information.

These translate to following goals for this dissertation:

1. **Modeling *en route* changes in lane choice:** Adaptive route choice under the presence of real-time information, also referred as online routing, has been well studied for general networks [18, 19, 20]. However, managed lanes networks are acyclic and thus efficient online routing algorithms can be adapted for these networks. Under this goal, we extend the existing models in the literature to managed lanes and compare the performance of currently used lane-choice models in the literature with an optimal online routing model.

2. **Developing improved dynamic pricing models for short-term operations:** By modeling the dynamic tolling as a Markov decision process, our goal is to propose a heuristic that generates toll prices which does better than the existing heuristics on two tolling objectives, incorporates adaptive lane choices made by a traveler, and handles tradeoff between optimizing for multiple objectives together.

3. **Modeling multiclass user-equilibrium under toll and travel time uncertainty**: When each traveler expects tolls and travel times to be stochastic, they route themselves using policies which minimize their expected costs. Under the presence of network congestion where there are several travelers using the managed lane network, we expect that travelers' choice of routing policies will converge to an equilibrium flow. We term this multiclass user equilibrium with recourse, where at equilibrium all travelers with a certain value of time (VOT) going from same origin to the same destination follow policies such that all used policies have equal and minimal expected costs. Under this goal, we study the multiclass static user equilibrium with recourse (M-UER) models for managed lane networks and conduct sensitivity analysis of the equilibrium for network contraction of managed lanes as part of the traditional equilibrium models.

1.3 Contributions

Figure 1.2 outlines the chapters in the dissertation and states the contribution for each chapter in a few words. The detailed contributions from each chapter are described as follows:

Single Driver Route Choice Model (Chap 2): First, we present a methodolog-

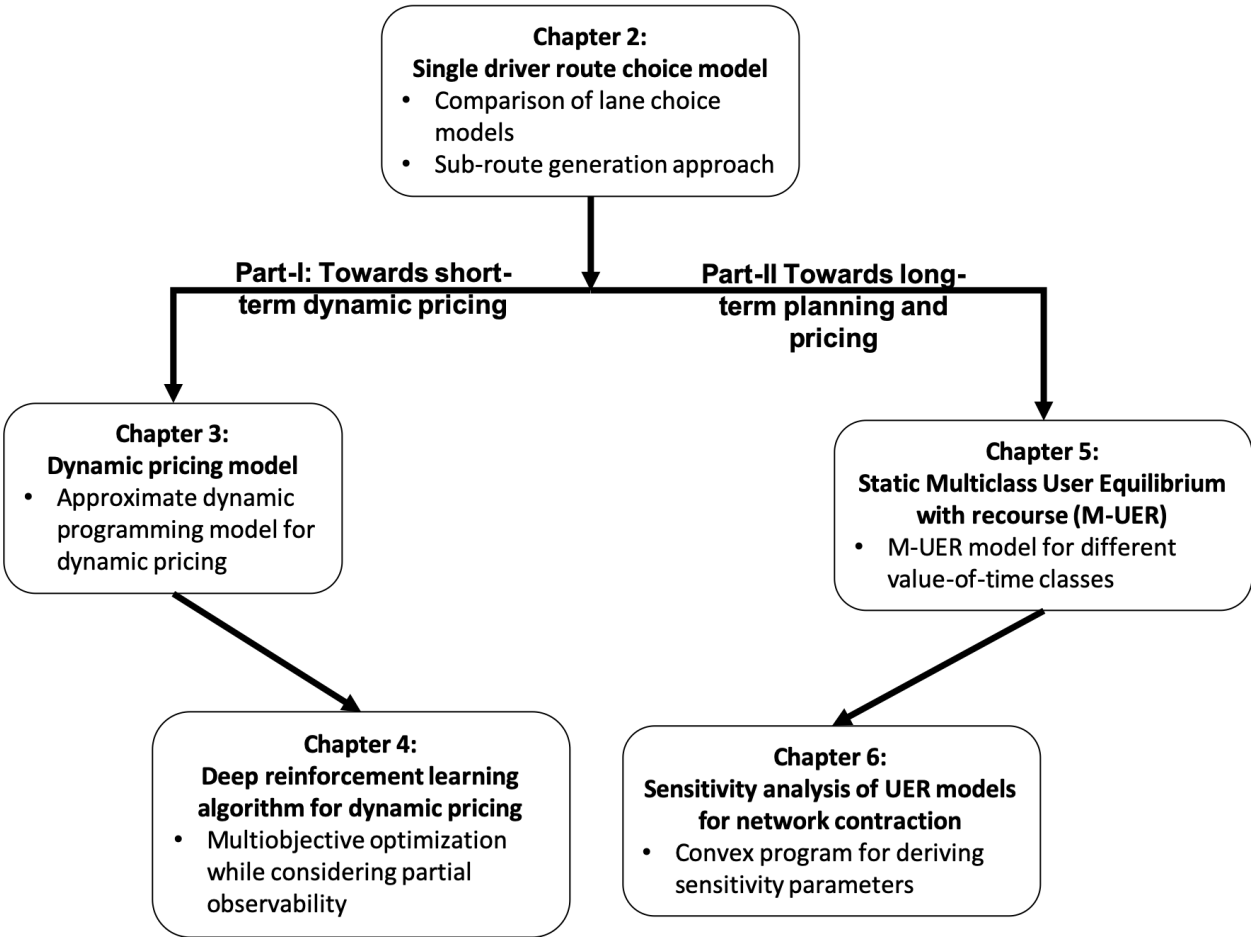


Figure 1.2: Organization of chapters and contribution

ical comparison of different route choice models and make recommendations on the choice of appropriate route choice model to develop efficient component models for managed lanes planning and operations. Second, we demonstrate how different assumptions on driver behavior produce variations in route choice, forming a basis for future research in understanding driver's lane choice patterns.

Dynamic Pricing Model for Short-term Operations (Chap 3): First, we develop a model where travelers make online decisions at each diverge point considering all routes on a managed lane network. The proposed method compares utility across a set of routes which grow quadratically with network size at each diverge point. Such online route choice on managed lane networks has not been studied in the literature to the best of our knowledge. Second, we demonstrate how the value function approximation (VFA) algorithm from the approximate dynamic programming (ADP) literature can determine tolls which

perform better than "greedy" tolling schemes and other heuristics used in practice. Third, we also develop a distributed model for dynamic pricing of managed lanes with multiple entrances and exits that scales for networks with multiple toll segments. And last, with an appropriate selection of the lane choice model, we develop a method to explore continuous toll action space for all agents which obviates the need to generate and evaluate discrete toll values like previously done in the literature.

Deep reinforcement learning algorithm for dynamic pricing (Chap 4): First, we demonstrate the usefulness of Deep-RL algorithms for solving dynamic pricing control problem under partial observability, and show that it performs well against existing heuristics, without requiring restricting assumptions on driver behavior or traffic dynamics. Second, we apply multi-objective optimization methods for joint optimization of multiple objectives and overcome undesirable jam-and-harvest (JAH) characteristics of revenue-maximizing optimal policies. Last, we conduct tests to verify the transferability of learned Deep-RL algorithms to new input distributions and make recommendations on real-time implementation of the algorithm.

Static Multiclass User Equilibrium with Recourse (M-UER) (Chap 5): We propose a multiclass formulation of the model for user-equilibrium with recourse where a traveler seeks to minimize a linear combination of two criteria: tolls and travel time.

Sensitivity analysis of UER for network contraction (Chap 6): First, we present a gradient-projection algorithm for generating computationally-efficient solutions to UER with better accuracy than the algorithms in the literature. And second, we present a convex program for sensitivity analysis of UER models and present an extension of the gradient-projection algorithm above for computing the sensitivity parameters.

1.4 Organization

The dissertation has been written so all the chapters can be read independent of each other. The rest of the document is organized as follows. Chapter 2 discusses the online route choice model for single traveler. Chapter 3 focuses on the dynamic pricing problem for short-term operations where a traveler only rely on the real-time information to make online decisions and the case where toll operations at each toll gantry are decentralized. Chapter 4

extends the dynamic pricing problem to settings where traffic state is not fully observable. Chapter 5 presents a static user-equilibrium with recourse framework for long-term planning of managed lanes. Chapter 6 presents a sensitivity analysis model for network contraction of acyclic express lane networks in the traditional user equilibrium models. Chapter 7 summarizes the findings in the dissertation and discusses ideas for future work.

Chapter 2

Single Driver Route Choice Model

[1]Modeling driver behavior and lane choice patterns is crucial for the success of several components of managed lane planning and operations, including traffic and revenue forecasting and toll pricing. A route choice model, also referred as a lane choice model in the literature, predicts decisions of travelers at each diverge location. The analysis of lane choice decisions is complex for managed lanes with multiple entrances and exits due to multiple decision locations. Improper assumptions on driver behavior may lead to incorrect traffic or revenue forecasts and may negatively impact reliability on the corridor. For example, a recent analysis of traffic speed data on the I-495 managed lanes in Washington showed that the minimum speed standards on the 15-mile-long managed lane corridor were met only 81% of the time [13].

Current route choice models for managed lanes with multiple entrances and exits make limiting assumptions, for example, travelers entering a managed lane do not exit until the destination is reached [10, 11] or that a traveler only relies on real-time information to make her decision [21]. With the influx of connected and autonomous vehicles, we expect that travelers (or vehicles) can soon learn the historic pattern and the evolution of traffic and toll prices and can adapt their routes dynamically (also referred to as online adaptation of routes). Under such cases, existing route choice models for managed lanes are not sufficient. Routing algorithms for networks with dynamic information have been a widely studied area of research [22, 23, 24]. However, managed lane networks which have an acyclic nature and a defined tolling architecture present a special testbed for adapting routing algorithms in the literature to understand driver behavior.

Furthermore, a recent data-driven analysis of lane choice on managed lanes found

[1]The chapter has been published as following:
a) Venktesh Pandey and Stephen D. Boyles. Comparing route choice models for managed lane networks with multiple entrances and exits. *Transportation Research Record*, 2673 (10):381-393, 2019.;
The contributions of Venktesh Pandey include study conception and design, conducting simulations, data collection, analysis and interpretation of results, and manuscript preparation.

that the existing route choice model predictions do not match well with field data [25]. Some travelers were found to make the same lane choice decision regardless of the toll and travel time information, while others were found to choose lanes inconsistent with their approximate value of time (VOT). This study shows the need for research which evaluates appropriate driver behavioral assumptions and their impacts on route choice of a traveler.

This chapter seeks to fill this gap in the literature. We formulate an online route choice model that determines the lane choice of a traveler at each diverge node in a managed lane network with multiple entrances and exits that minimizes the total expected cost. We compare the performance of the formulated model against other routing methods in the literature and demonstrate the impact on results due to variations in driver behavior. We also make recommendations on the choice of routing models when embedded within other planning models for managed lanes.

The primary contributions of this work are two-fold. First, we present a methodological comparison of different route choice models and make recommendations on the choice of appropriate route choice model to develop efficient component models for managed lanes planning and operations. Second, we demonstrate how different assumptions on driver behavior produce variations in route choice, forming a basis for future research in understanding driver's lane choice patterns.

2.1 Literature review

The literature on lane choice models for managed lanes with a single entrance and exit can be classified as binary logit, VOT distribution, and all-or-nothing assignment [26]. The most commonly-used binary logit model predicts the probability of choosing a managed lane given the toll and travel time difference between the two lanes [27, 28]. On the other hand, models based on VOT distribution attribute the differences in traveler choices to the variability in their VOT values instead of an error associated with the perception of the utility of an alternative. Gardner et al. [26] show that a logit model may cause inconsistent behavior such as travelers choosing managed lanes even if there is no congestion, which a VOT distribution-based approach avoids. The all-or-nothing assignment is a special case of VOT distribution when the distribution is uniform.

In a recent data-driven analysis for the managed lanes in Katy, TX and North Tarrant Expressway (Dallas-Fort Worth), Burris and Brady [25] observe that travelers often choose a certain lane regardless of the toll or travel time values. They also find that travelers are not really observed to optimize their lane choice and have shown to take choices which are inconsistent with their past behavior. Under limited information settings, we can expect drivers to behave suboptimally. However, in the near future, smart devices in a connected vehicle can collect historic and real-time data and direct travelers towards an optimal decision at each diverge. In such cases, we can expect travelers to make optimal decisions. Our research includes analysis for these scenarios where a traveler has navigated the managed lane system enough to learn about the network conditions and its variation.

Developing pricing or planning models for managed lanes with multiple entrances and exits has been a recent field of research where the assumptions made on route choice are simplistic. Yang et al. [11] solve for optimal tolls for each entrance and exit by assuming that if a traveler enters the managed lane, they do not exit it until their destination. Zhu and Ukkusuri [10] and Tan and Gao [12] make similar assumptions in their pricing models and use a binary logit model to determine lane choice. Pandey and Boyles [21] use VOT distributions to model lane choice by comparing utilities over a set of routes called decision routes. These current models are built on the use of real-time information for dynamic routing and fail to consider the availability of historic information that a traveler may learn from experience.

This chapter focuses on routing vehicles on stochastic time-varying (STV) managed lane networks, an area which has been extensively studied in more general networks. There are two broad categories in this area: one finding the least-expected-cost path before starting a route and the other finding a least-expected-cost strategy that makes changes to the route as it is traversed based on the real-time information.

Finding a least-expected-cost path in STV networks is an NP-hard problem [29], as Bellman's principle of optimality does not hold true due to the non-linear nature of the expectation operator. Miller-Hooks and Mahmassani [24] proposed an algorithm that finds "non-dominated" paths from all nodes to the destination. Several extensions to this work and other heuristic algorithms have been proposed. Readers are referred to Prakash [30] for

a review of these algorithms.

A least-expected-cost strategy in STV networks chooses a link downstream of a node to minimize the expected cost given the current information. Since Bellman's principle of optimality holds in this case [29], polynomial and pseudopolynomial algorithms have been developed for finding optimal routing policy [19, 31]. Gao and Chabini [22] present a general framework for finding optimal routes when link travel times are correlated. These algorithms have been extended for network design problems like finding the optimal location of variable message signs (VMSs) [32].

Optimal online routing strategies for optimizing multiple criteria have also been studied. If we assume that travelers seek to minimize a linear combination of tolls and travel time, the algorithms from the single-criterion optimization can be easily extended to solve multi-criteria problems. Jafari and Boyles [33] use a backward recursion algorithm for finding online shortest paths for an electric vehicle while minimizing its wait time at a charging station, travel time, and cost associated with charging. Opasanon and Miller-Hooks [23] present a general framework for solving the least-expected-cost strategy for multiple criteria and discuss algorithms that determine Pareto-optimal "hyperpaths". Models for managed lanes with reliability as a component of lane choice have also been studied [25]. Prakash and Srinivasan [34] present a formulation based on "hypergraphs" for solving robust optimal strategies which minimizes the combination of mean travel time and standard deviation, which is a manner of expressing reliability.

This research formulates the online routing problem as a Markov decision process and uses a backward recursion algorithm to solve the problem on managed lane networks as a special case from the current literature. We also compare the online routing problem results with the other routing models in the literature.

2.2 Routing models

This section explains the routing model for a single vehicle in STV managed lane networks. We first present the notation and framework for our analysis, then discuss the formulation for optimal route choice model, and end the section with a discussion of other routing models. Throughout the text, we use vehicle and traveler interchangeably.

2.2.1 Notation and framework

This section introduces the notation related to different components of the formulation. Figure 2.1 shows an abstract network for managed lanes, where the top set of links form the managed lanes (ML) while the bottom set of links form the general purpose lanes (GPL). There are three entrances to the managed lane and two exits out of the managed lane as shown.

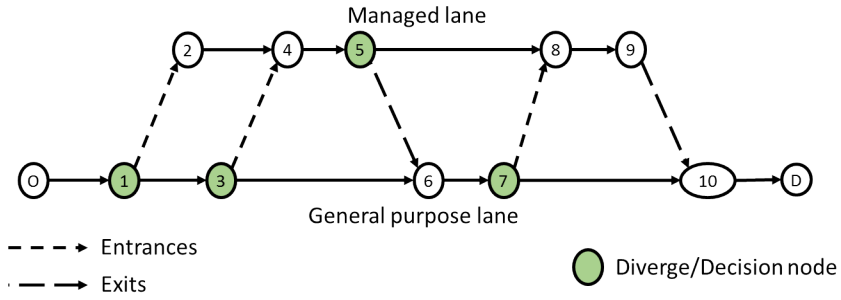


Figure 2.1: An abstract managed lane network

Network

Consider a directed acyclic graph for managed lanes $G = (N, A)$, where N and A are the sets of nodes and links, respectively. A link $(i, j) \in A$ connects nodes i and j, where $i, j \in N$. Let $\Gamma(i)$ and $\Gamma^{-1}(i)$ represent the set of outgoing and incoming links from and to node i, respectively. Let $\mathscr{T} = \{0, 1, 2, \dots T\}$ denote the discrete set of time intervals in our period of interest. We model route choice of a single vehicle traveling from origin $o \in N$ to destination $d \in N$, but other vehicles may be present. The route choice model of a single vehicle is analogous to finding shortest path while solving user equilibrium on transportation networks. Equilibrium analysis of routing for multiple vehicles is part of the future work. Let $N_{\text{div}} \subset N$ denote the set of diverge nodes where the vehicle makes a decision. These nodes are highlighted in Figure 2.1.

Tolling

Several options have been suggested in the literature for charging a traveler for accessing a managed lane [1, 35]. We briefly discuss each of the toll options. Each tolling option can be charged either as a fixed or distance-based rate at any given time step.

1. **Option 1–Tolling by segment/zone**: In this option, the managed lane network is divided into multiple segments and a traveler pays the same toll if they enter and exit the managed lane anywhere within that segment. An additional toll is charged if a new segment is entered. For example, for the network in Figure 2.1, say that links $(2, 4)$ and $(4, 5)$ belong to segment 1, and links $(5, 8)$ and $(8, 9)$ belong to segment 2. A traveler entering at 2 and exiting at 4 will pay the rate for segment 1, while a traveler entering at 2 and exiting at 9 will pay the rate for both the segments.

2. **Option 2–Tolling by entrance**: In this option, a traveler entering the ML at any given entrance pays the same toll rate regardless of the exit. For the network in Figure 2.1, the toll rate is different if a vehicle enters at $2, 4$, or 8. Option 1 can be developed as a special case of this option.

3. **Option 3–Tolling by entrance-exit pair**: In this option, a traveler pays a specific toll rate based on the entrance and the exit location from the managed lane. There is no difference between a fixed rate and distance-based rate for this option since the distance between an entrance and exit is fixed. Options 1 and 2 can both be developed as a special case of this option.

In this chapter, we choose another tolling option which we call **option 4–tolling by diverge location**. Here a traveler pays a separate toll at every diverge node $i \in N_{\text{div}}$. For the network in Figure 2.1, a certain rate is charged if a traveler enters the managed lane at node 1. If the traveler continues to travel on the managed lane at node 5, they pay an additional rate. Toll options 1 and 2 are special cases of this option. Toll option 3 offers more degrees of freedom in controlling tolls; however, we must employ restrictions on toll possibilities in option 3 to ensure fairness for the charged tolls, e.g. the toll for $(1, 2) \rightarrow (5, 6)$ entrance-exit pair should be lower than the toll for $(1, 2) \rightarrow (9, 10)$.

We choose option 4 for the simpler models it provides. This option allows for tolls to be collected on all ML arcs directly succeeding a diverge node. Let A_{toll} be the links where tolls are charged. For the network in Figure 2.1, $A_{\text{toll}} = \{(1, 2), (3, 4), (5, 8), (7, 8)\}$. Let $\tau_{ij}(\cdot)$ denote the dynamic toll charged on link $(i, j) \in A_{\text{toll}}$. We set $\tau_{ij}(\cdot) = 0$ for all links $(i, j) \in A \backslash A_{\text{toll}}$. We explain the time dependence of tolls later in this section.

Driver behavior

Let α denote the VOT of a vehicle being routed from the origin node $o \in N$ to the destination node $d \in N$. For our purposes, a rational traveler always seeks to minimize their cost, while an irrational traveler may choose lanes based on a certain preference. We model an irrational traveler using parameter $\epsilon_r \in [0, 1]$, which models the probability that the traveler will behave rationally. We define another variable, ϵ_{ML}, which indicates the conditional probability that a traveler prefers the managed lane given that a traveler behaves irrationally ($\epsilon_r < 1$). If $\epsilon_r < 1$, then ϵ_{ML} may range between $[0, 1]$, where a value of 0 (1) indicates that the traveler always chooses the GPL (ML). It is possible that a traveler with $\epsilon_r < 1$ may end up making the same decision as a traveler who always makes decisions rationally [25].

Stochasticity and dynamic information

We model arc travel times and tolls as stochastic, time-varying, finite, and discrete random variables. In our current model, we assume tolls and travel times are independent of each other. This assumption is made to simplify the modeling process but may not hold in reality, where tolls are set based on the evolution of travel times, and travel times on the links have spatial and temporal correlation. This is similar to the independence assumptions in past literature [23, 31, 33]. In our future work, we will relax this assumption.

We further assume that upon arrival at any node in the network, each traveler learns about the toll and travel time realizations on downstream links. There can be alternate ways of learning information about tolls and travel time, example using VMSs or traveler information systems. However, several of the current implementations of managed lanes only display tolls right before the diverge location. In such settings, it is reasonable to assume that a traveler learns the toll and can see the congestion on the downstream links immediately ahead of her. In our future work, we will extend our analysis to alternate assumptions on information acquisition.

Let $\Theta_i(t)$ denote the set of all possible informationvectors that can be made available to a traveler at node i at time step t, where an information vector $\theta \in \Theta_i(t)$ consists of travel time and toll values of the downstream arcs. Let $t_{ij}(\theta)$ and $\tau_{ij}(\theta)$ denote the travel time and toll values on link (i, j) for information vector $\theta \in \Theta_i(t)$. The probability of occurence of

information vector $\theta \in \Theta_i(t)$ is denoted by $v_i^\theta(t)$ and is assumed to be known to the traveler in advance. This is a reasonable assumption for experienced drivers or for scenarios in the future with connected or autonomous vehicles which can track information across days.

2.2.2 Optimal online route choice model

The optimal routing seeks to minimize the total expected cost after departing the origin at a certain time. There are two criteria used in routing, travel time and toll. We simplify the modeling process by assuming that the total cost is a linear combination of toll and travel time, thus the algorithms from the single criterion literature apply.

We formulate the problem as a non-discounted finite horizon Markov decision process (MDP) with a termination state. MDPs are a traditional method for solving problems involving stochasticity and dynamic decision-making. The components of the MDP are defined as following:

1. **State space**: The current state of a vehicle is given by $\boldsymbol{s}_{\text{curr}}$, defined as $\boldsymbol{s}_{\text{curr}} = (i_{\text{curr}}, t_{\text{curr}}, \theta_{\text{curr}})$, where i_{curr} is the current node location of the vehicle, t_{curr} is the current time step, and $\theta_{\text{curr}} \in \Theta_{i_{\text{curr}}}(t_{\text{curr}})$ is the current information received at node i_{curr}. The simulation terminates when the vehicle has arrived at the destination. We define termination state set $\mathscr{C} = \{\boldsymbol{s}_{\text{curr}} \mid i_{\text{curr}} = d\}$, where $d \in N$ is the destination node of the vehicle.

2. **Action space**: The action space given a state $\boldsymbol{s}_{\text{curr}}$ is the set of downstream links from the current node. We denote the action space by $U(\boldsymbol{s}_{\text{curr}})$ which is same as the set $\Gamma(i_{\text{curr}})$. Let $a \in U(\boldsymbol{s}_{\text{curr}})$ be an action in the action space representing a link in the network and $\texttt{headNode}(a)$ represent the head node of the link.

3. **Transition function**: Given a state $\boldsymbol{s}_{\text{curr}}$ and the action $a \in U(\boldsymbol{s}_{\text{curr}})$, the transition function $f(\boldsymbol{s}_{\text{curr}}, a)$ determines the next state $\boldsymbol{s}_{\text{next}}$ visited by the vehicle. The evolution

dynamics of $f(\cdot)$ are given in Equation (2.1):

$$\begin{aligned} i_{\text{next}} &= \texttt{headNode}(a) \\ t_{\text{next}} &= t_{\text{curr}} + t_a(\theta_{\text{curr}}) \\ \theta_{\text{next}} &= \text{any } \theta \in \Theta_{i_{\text{next}}}(t_{\text{next}}) \text{ with probability } v^{\theta}_{i_{\text{next}}}(t_{\text{next}}) \end{aligned} \tag{2.1}$$

4. **One step cost**: Given a state $\boldsymbol{s}_{\text{curr}}$ and the action $a \in U(\boldsymbol{s}_{\text{curr}})$, the one step cost is given by $g(\boldsymbol{s}_{\text{curr}}, a) = \alpha t_a(\theta_{\text{curr}}) + \tau_a(\theta_{\text{curr}})$

An adaptive routing policy $\mu(\cdot)$ determines the action a to be taken in each state $\boldsymbol{s}_{\text{curr}}$ at all time steps. The objective of solving the MDP is to determine an adaptive routing policy that minimizes the expected cost. The expected cost-to-go starting from state $\boldsymbol{s}_{\text{curr}}$ and using a certain policy μ is given by $J_\mu(\boldsymbol{s}_{\text{curr}})$ which is defined as shown in Equation (2.2).

$$J_\mu(\boldsymbol{s}_{\text{curr}}) = E[g(\boldsymbol{s}_{\text{curr}}, \mu(\boldsymbol{s}_{\text{curr}})) + J_\mu(f(\boldsymbol{s}_{\text{curr}}, \mu(\boldsymbol{s}_{\text{curr}})))] \tag{2.2}$$

By definition of termination state, $J_\mu(\boldsymbol{s}_{\text{curr}}) = 0$ for all $\boldsymbol{s}_{\text{curr}} \in \mathscr{C}$.The optimal routing policy μ^* is the one that minimizes the total expected cost over all policies. The cost-to-go values in any state when following the optimal policy satisfy the Bellman equation as shown in Equation (2.3). We also denote the optimal cost-to-go value of being in state $\boldsymbol{s}_{\text{curr}}$ as $J^*(i_{\text{curr}}, t_{\text{curr}}, \theta_{\text{curr}})$.

$$J^*(i_{\text{curr}}, t_{\text{curr}}, \theta_{\text{curr}}) = J_{\mu^*}(\boldsymbol{s}_{\text{curr}}) = \min_{\forall a \in U(\boldsymbol{s}_{\text{curr}})} g(\boldsymbol{s}_{\text{curr}}, a) + E[J_{\mu^*}(\boldsymbol{s}_{\text{next}})] \tag{2.3}$$

where $E[J_{\mu^*}(\boldsymbol{s}_{\text{next}})]$ is the expectation over all possible next states reached by taking an action a in the current state.

Finding the optimal cost-to-go values for each node is the new objective as knowing these can determine the optimal policy. This finite-horizon MDP can be solved using the standard backward recursion algorithm shown in Algorithm 1. The algorithm starts with an initialization of the cost-to-go values for each state and improves them sequentially by going in a reverse topological order from the destination node. The algorithm exploits the acyclic

nature of the managed lane network and terminates with optimal cost-to-go values in one sweep of the network. We denote this route choice model by `OSP` in the remaining text.

Algorithm 1 Backward recursion algorithm for solving the proposed MDP

```
Initialization
for t ∈ 𝒯 do
    Set J*(d, t, θ) = 0 ∀θ ∈ Θ_d(t)
    Set J*_i(i, t, θ) = ∞ ∀i ∈ N\d, ∀θ ∈ Θ_i(t)
end for

Update the value
for node i in reverse topological order do
    for t ∈ 𝒯 do
        for information θ ∈ Θ_i(t) do
            Update cost for J*(i, t, θ) using Equation (2.3)
        end for
    end for
end for
```

2.2.3 Other routing models

In this section, we discuss other routing models proposed in the literature focusing in particular on models for managed lanes with multiple entrances and exits. We first define the notion of instantaneous and experienced cost for any path in the network.

We define a path (or route) $\pi = [j_0, j_1, \ldots, j_{K_\pi}]$ in the network as an ordered set of nodes $j_k \in N$ for all $k \in \{0, 1, \ldots, K_\pi\}$, where K_π is the number of links in the path. Let $\pi(i) = (j_{i-1}, j_i)$ be the ith link in the path, $i \in \{1, \ldots, K_\pi\}$. Furthermore, let $\hat{\theta}_i(t) \in \Theta_i(t)$ be the realization of the travel time and toll information at node $i \in N$ at time step t. The instantaneous cost of path π at time step t is defined as the sum of total cost for each link on the path using the realizations of travel time and tolls at the current instant of time. It is denoted by $U_\pi^{\text{inst}}(t)$ and is calculated using Equation (2.4), where the summation is over all links in the path and α is the VOT value of the traveler.

$$U_\pi^{\text{inst}}(t) = \sum_{i=1}^{K_\pi} \tau_{\pi(i)}(\hat{\theta}_{j_{i-1}}(t)) + \alpha t_{\pi(i)}(\hat{\theta}_{j_{i-1}}(t)) \tag{2.4}$$

The experienced cost of a path is defined similarly, as the sum of costs for each

link on the path. However, it takes into account the time of arrival at future nodes. We define the expected experienced cost of path π at time step t recursively by calculation expectation over all possible realizations of travel time and toll in the future. We denote this by $U_{\pi}^{\exp}(t)$ and calculate it using Equation (2.5), where α is the VOT value of the traveler and $\bar{\pi} = [j_1, j_2, \ldots, j_{K_{\bar{\pi}}}]$ is the path obtained by removing the first node (or link) from the current path π.

$$U_{\pi}^{\exp}(t) = \sum_{\theta \in \Theta_{j_0}(t)} \left(\tau_{\pi(1)}(\theta) + \alpha t_{\pi(1)}(\theta) + U_{\bar{\pi}}^{\exp}(t + t_{\pi(1)}(\theta))\right) v_{j_0}(\theta) \tag{2.5}$$

Binary logit model

Binary logit route choice models used in the literature [10, 11, 12] assume that travelers do not exit the managed lane after they enter, until the end of the corridor. The diverge nodes located on the managed lanes are no longer considered as decision point. At each diverge node along GPL, a traveler compares instantaneous utility between two paths: one connecting the current node to the destination only using links on ML while the other connects the current node to the destination only using links on GPL.

Figure 2.2(a) shows routes over which the utility is compared at each diverge node. We denote the path using the managed lane by π_{ML} and the path using the general purpose lane by π_{GPL}, where the first node on both paths is the current diverge node j_0. The route choice decision is made by computing the probability of choosing the managed lane p_{ML} given the current realizations of toll and travel time at any diverge node. The probability p_{ML} is evaluated using Equation (2.6), where ζ is the inverse of the scale parameter for a logit model which can be used to calibrate the model. We denote this route choice model by `Logit` in the later text.

$$p_{\mathrm{ML}} = \frac{e^{-\zeta U_{\pi_{\mathrm{ML}}}^{\mathrm{inst}}(t)}}{e^{-\zeta U_{\pi_{\mathrm{ML}}}^{\mathrm{inst}}(t)} + e^{-\zeta U_{\pi_{\mathrm{GPL}}}^{\mathrm{inst}}(t)}} \tag{2.6}$$

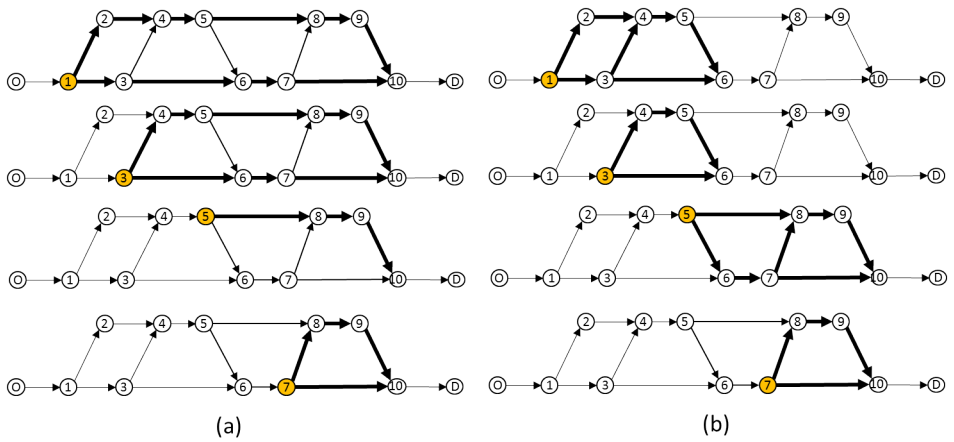


Figure 2.2: Routes for which the instantaneous costs are compared: (a) `Logit` *model and (b)* `Decision Route` *model*

Decision routes based route choice model

Proposed in Pandey and Boyles [21], this approach considers a set of routes, called decision routes, at each diverge location on the managed lane network. Intuitively, at each diverge node, a decision route connects the current node with the head node of the first exit from the ML downstream of the current node entrance. This approach avoids enumeration of all possible routes to the destination and maintains scalability of the model. A formal definition is given in [21]. Figure 2.2(b) shows the decision routes for each diverge node. Let Π^i_{DR} denote the set of decision routes at any diverge node $i \in N_{\text{div}}$.

At each diverge node, the traveler makes an online decision by comparing the instantaneous costs across each decision route and choosing the first link of the path minimizing the cost. Formally, for a given diverge node $i \in N_{\text{div}}$, the chosen link (i, j) is given by Equation (2.7). We denote this route choice model by `Decision Route` in the later text.

$$\begin{aligned} \pi^* &= \text{argmin}_{\pi \in \Pi^i_{\text{DR}}} U^{\text{inst}}_{\pi}(t) \\ (i, j) &= \pi^*(1) \end{aligned} \tag{2.7}$$

Offline route choice model

Under this routing model, a complete route is selected before the vehicle departs from the origin and the path is not adapted later. We use the concept of Pareto-optimal paths as discussed in Miller-Hooks and Mahmassani [24]. A path π is called a dominated path

if there is at least one path π' connecting the same first node and the last node such that $U^{\exp}_{\pi'}(t) \leq U^{\exp}_{\pi}(t)$ for all $t \in \mathscr{T}$, and $U^{\exp}_{\pi'}(t) < U^{\exp}_{\pi}(t)$ for at least one $t \in \mathscr{T}$.

A set $P(i)$ of Pareto-optimal paths between nodes i and the destination d of the vehicle is defined as a set of paths which are not dominated. We adapt the algorithm in Miller-Hooks and Mahmassani [24] and Jafari and Boyles [33] for solving Pareto-optimal paths for all nodes as shown in Algorithm 2. The Pareto-optimal path from origin o at a given departure time is the path that minimizes the expected cost to the destination. We denote this route choice model by `Offline` in the later text.

Algorithm 2 Offline routing algorithm

```
Create an artificial path π from d to d
for t ∈ T do
    Set U^exp_π(t) = 0
end for
SEL ← π
while SEL ≠ Φ do
    Remove path π from SEL. Let j be the path origin.
    for i ∈ Γ^{-1}(j) do
        Define η = (i, j) ⊕ π
        For all t ∈ T, find U^exp_η(t) using Equation (2.5)
        If η is Pareto-optimal, then P(i) ← P(i) ∪ η and SEL ← SEL ∪η
        Remove set of paths from dominated by η from P(i) and SEL
    end for
end while
```

Random policy

This routing policy chooses the next link at each decision node randomly with equal probability and is denoted by `Random` in the later text. This policy is included as a baseline where a traveler uses no historic or real-time information in choosing her routes.

Each of these routing models can be categorized based on the type of historic and real-time information used in deciding the route. Table 2.1 shows the taxonomy of the routing models discussed above. The `Logit` and `Decision Route` models only use real-time information on the routes over which the instantaneous utilities are compared. The `Offline` model uses only the historic information on probability distributions of tolls and travel time. The `OSP` model uses both information sources for determining the optimal route, while the `Random` model uses none.

Table 2.1: Taxonomy of route choice models considered in this study

Route choice model	Information used by the route choice model	
	Historic information from past experience	Real time (online) information on VMSs
`OSP`	Probability distribution of travel time and toll for each link	Travel time and toll information on the downstream arcs at a given node
`Logit`	NONE	Travel time and toll information along two routes from the current node to the destination (one using the ML and other using the GPL)
`Decision Route`	NONE	Travel time and toll information along each path in the decision route set
`Offline`	Probability distribution of travel time and toll for each link	NONE
`Random`	NONE	NONE

In the next section, we compare the performance of the discussed routing algorithms on different STV managed lane networks. Since we make the assumption that travel time and tolls are independent spatially and temporally, the differences in the information requirements among `DR`, `Logit`, and `OSP` model, shown in Table 2.1, still make for a valid comparison. If the travel times are correlated, the differences in information will impact the shortest path cost; this is left as future work.

2.3 Experiments

We compare the route choice models discussed in Section 2.2 on three test networks: a double-entrance-single-exit (DESE) network, the LBJ TEXpress network consisting of three entrances and two exits (LBJ), and a network with thirteen entrances and fourteen exits (13En14Ex). The DESE and 13En14Ex networks are constructed artificially, where the length of each entrance and exit ramp is set as 0.15km, and the length of other links is a multiple of this length, with multipliers randomly sampled from the set $\{2, 3, 4\}$. The LBJ network is constructed using the lengths for the first toll segment from the LBJ TEXpress network in Dallas, TX. Figure 2.3 shows an abstraction of these networks.

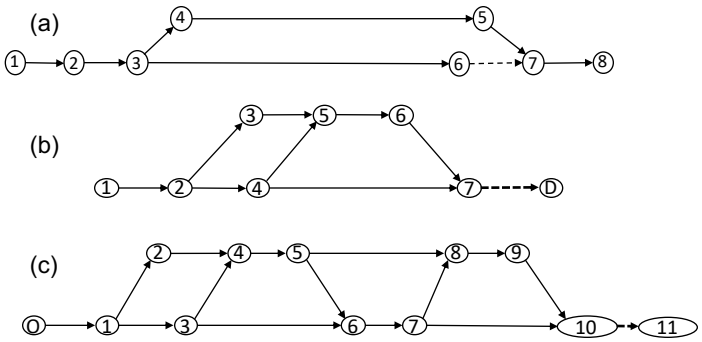


Figure 2.3: Networks used for analysis: (a) Double-entrance-single-exit network (DESE), (b) LBJ TEXpress network (LBJ), and (c) 13 entrance 14 exit network (13En14Ex)

The free-flow travel time $\bar{t}_{ij}$ for each link $(i,j) \in A$ is determined by dividing the length by its free-flow speed, set as 90 km/hr for all links on GPL and 120 km/hr for all links on ML. The travel times for each link on ML is sampled from the set $\{\bar{t}_{ij}, 1.1\bar{t}_{ij}\}$, while travel times for each link on GPL is sampled from the set $\{m\bar{t}_{ij}\}$, where m takes value between 1 and 2 in increments of 0.1. The toll on each link $(i,j) \in A_{\text{toll}}$ is sampled randomly from the set $\{\$0.1, \$0.3, \$0.5, \$0.7, \$0.9\}$, while tolls on other links are set at \$0. The possible combinations of toll and travel time information on each link are generated and each combination is assigned the same probability of occurrence. Each routing model was implemented in Java on a 3.3 GHz Windows machine with 4 GB RAM.

2.3.1 Rational traveler

The analysis in this section considers a traveler with $\epsilon_r = 1$ for the `OSP` model, that is, the traveler always follows the optimal policy action predicted by the model.

Figures 2.4(a) and (b) show the expected cost between vehicle's origin and destination obtained using the `OSP` model on the three networks for varying VOT values for the vehicle. The expected cost is expressed both in money units (\$) and time units, the latter obtained after dividing the money units by traveler's VOT. We observe that the expected cost is higher for vehicles with higher VOT, since, for a given value of travel time and toll on a link, a vehicle with higher VOT incurs higher costs. The costs are higher for network with more links because the average path length is longer. Decreasing expected cost in time units shows that the average travel time spent in the network is lower for vehicles with higher

VOT since they save time by traveling on the managed lanes in-spite of paying a toll. The DESE network shows an almost linear variation in the costs with a negligible slope which is possible if the difference between the best and the second best policy is large and all VOTs follow the same policy.

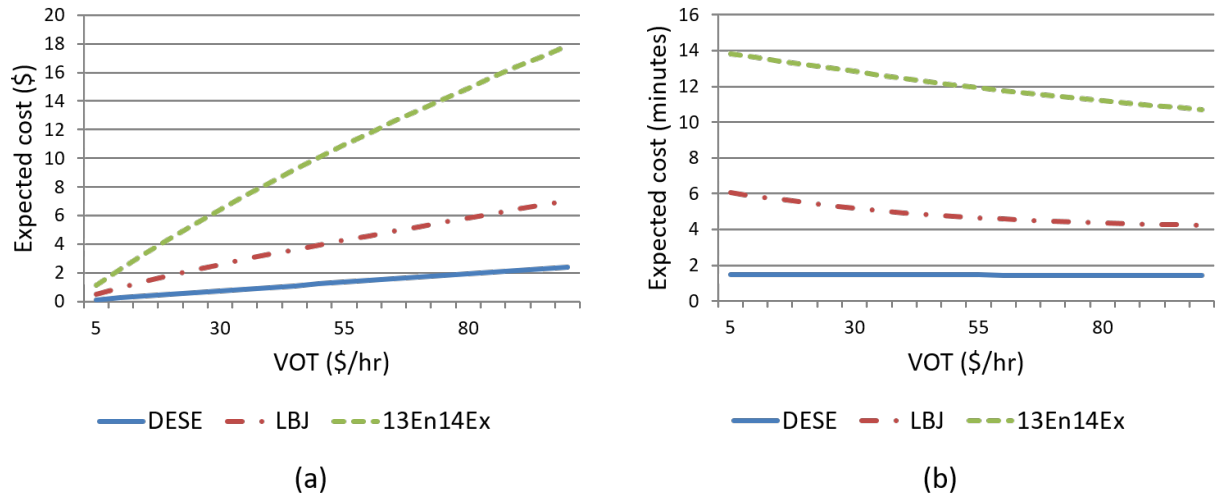


Figure 2.4: Comparison of expected costs in (a) dollar units and (b) travel time units

We also compare the expected costs of the shortest route using the `OSP` and the `Offline` models. Figure 2.5(a) shows the percent difference in costs from the `Offline` model measured relative to the `OSP` model and Figure 2.5(b) shows the difference in costs expressed in travel time units by dividing the difference by the VOT. The differences in costs between the two models is indicative of the potential benefit of providing online information in lowering the expected cost of a traveler. Expressed in units of time, we call this difference the value of online information (VoOI). This definition is similar to the one in the literature [36].

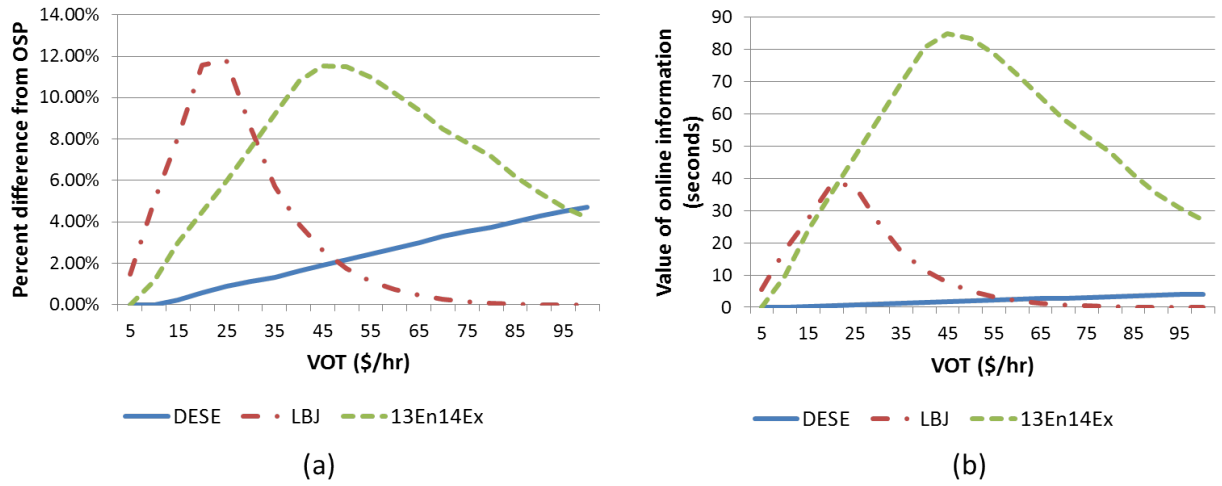


Figure 2.5: Comparison of the (a) percent difference in costs between the `Offline` *and the* `OSP` *model, and (b) value of online information for the three networks with varying values of time*

For the LBJ and 13En14Ex networks, VoOI value initially increases, then attains a peak, and decreases from then onwards. The location of the peak varies with the network. This pattern is expected: for vehicles with very low VOT, GPL is always preferable and the preference remains unchanged irrespective of the online information. Similarly, for vehicles with very high VOT, ML is always preferable and the presence of online information does not impact the preference. The benefit of online information is higher for vehicles with a certain VOT value. The peak VoOI for the LBJ and 13En13Ex networks are found to be 38 seconds and 85 seconds, respectively. For the DESE network, we do not observe this pattern over the simulated VOT values. This is because, for the generated network instance, the ML never becomes attractive as the toll values in the set are very high.

Next, we compare the expected costs obtained from simulating $100,000$ random information instances for all of the proposed methods for varying values of time. We plot the results for four VOT values ranging from \$15/hr to \$45/hr in increments of \$10/hr. Figure 2.6 shows the plots of the percent difference in expected cost for each model defined in Section 2.2.3 versus the `OSP` model. We test the `Logit` model for two values of ζ parameter, 0.1 and 0.01.

The percent error was found lowest for the `Decision Route` model, ranging between $0\% - 1.5\%$ for all three networks, with an average percent error of 0.93%. This suggests

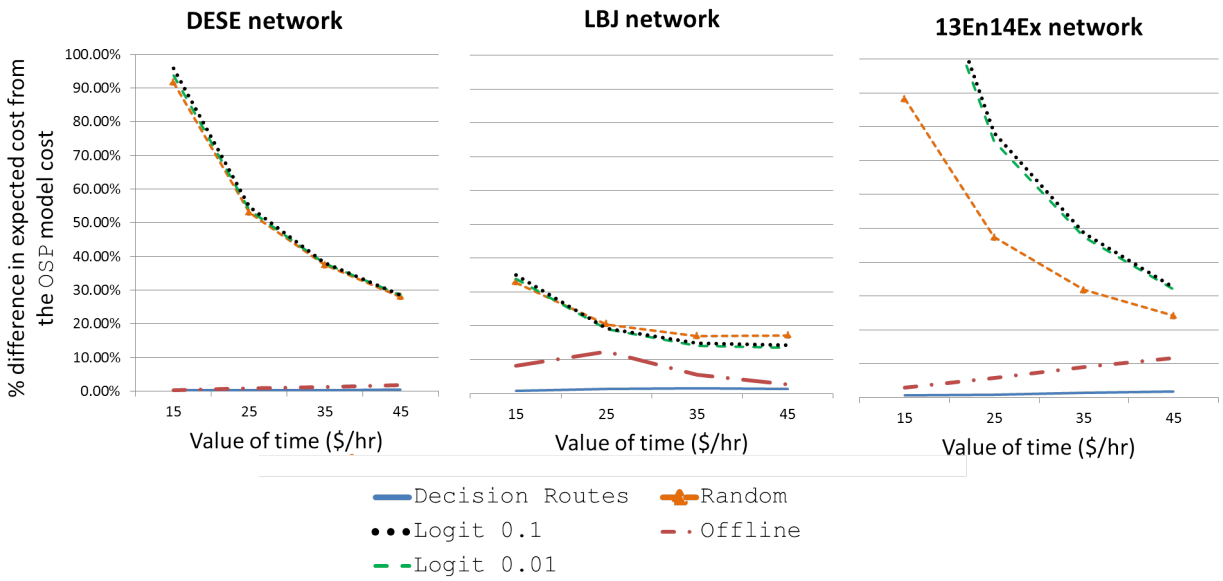


Figure 2.6: Comparison of percent difference in the expected cost between each of the four route choice model and the `OSP` *model for the three networks*

usefulness of the `Decision Route` model in determining online routes which are close to the optimal. The model is easy to implement as it relies only on the instantaneous cost predictions on decision routes and is thus relevant for use in the current planning or pricing models.

The error in the `Offline` model varied between $0\% - 12\%$ with patterns discussed earlier for the value of online information. We note that the benefit of the historic information used in the `Offline` model can be significant if travel times and tolls are highly correlated spatially and temporally, lowering the percent errors. In such settings, there can be a limited value in learning from the real-time information at diverge nodes, making offline routing beneficial [36]. Our modeling assumption that the link travel times and tolls are independent enhances the success of the `Decision Route` model compared to `Offline` model. Though models with correlation are much harder to solve, we will test the benefits of both models under correlation in future work.

We also observe that the `Logit` model for both values of ζ parameter performed almost identically. The model had an average percent error of 49.83% for $\zeta = 0.01$ and 51.09% for $\zeta = 0.1$, which is even higher than the average percent error of 40.76% for the `Random` model. The higher error in the `Logit` model shows its non-optimal prediction of lane

choice. This is reasonable since we use restrictions on exits from the ML in the definition of `Logit` model. The results show that making this assumption can have a significant impact in the accuracy of a driver behavior model if the traveler behaves rationally.

We also observe a trend that the error in the `Logit` model decreases with the increasing VOT value. This may be a result of the model preferring the managed lane when choosing routes. This preference leads to higher error in the expected cost for vehicles with lower values of time. In the next section, we compare `Logit` model for irrational travel behavior.

2.3.2 Irrational traveler

The analysis in this section assumes that a traveler makes sub-optimal decisions and the value of ϵ_r parameter is strictly less than 1. We model this behavior by generating 100,000 random information vectors and modeling that the traveler chooses actions predicted by the optimal policy with probability ϵ_r and a random action towards ML with probability $(1-\epsilon_r)\epsilon_{\text{ML}}$ and towards GPL with probability $(1-\epsilon_r)(1-\epsilon_{\text{ML}})$. We only compare the `OSP` model with $\epsilon_r < 1$ and the `Logit` model with varying values of ζ. Other models do not have an inherent structure for irrational route choice and are thus ignored for this analysis. We focus our analysis on the LBJ network for two values of time, \$20/hr and \$45/hr.

We compare the expected costs obtained for different values of time for varying values of ϵ_r and ϵ_{ML} parameters. Table 2.2 shows the variation of expected cost from the `OSP` model for increasing ϵ_r value on the vertical axis and increasing value of ϵ_{ML} on the horizontal axis. The heat map in the table shades a cell darker if the expected cost is higher relative to the value in other cells.

Table 2.2: Comparison of expected costs for varying values of ϵ_r and ϵ_{ML}

VOT = $20/hr			Increasing preference for choosing managed lane (ϵ_{ML})				
			Strong preference for GPL				Strong preference for ML
			0.001	0.3	0.5	0.7	0.999
Increasing probability of being rational (ϵ_r)	Highly random behavior	0.001	2.050	2.229	2.265	2.284	2.210
		0.1	2.025	2.178	2.222	2.237	2.202
		0.3	1.987	2.094	2.139	2.179	2.159
		0.5	1.950	2.025	2.062	2.101	2.104
		0.7	1.900	1.950	1.979	1.987	2.024
		0.9	1.863	1.874	1.881	1.897	1.899
	Highly optimal behavior	0.999	1.837	1.838	1.837	1.839	1.838

VOT = $45/hr			Increasing preference for choosing managed lane (ϵ_{ML})				
			Strong preference for GPL				Strong preference for ML
			0.001	0.3	0.5	0.7	0.999
Increasing probability of being rational (ϵ_r)	Highly random behavior	0.001	4.611	4.421	4.225	4.018	3.723
		0.1	4.520	4.341	4.161	4.013	3.720
		0.3	4.321	4.144	4.027	3.895	3.714
		0.5	4.121	4.001	3.910	3.825	3.702
		0.7	3.894	3.863	3.784	3.763	3.678
		0.9	3.711	3.688	3.676	3.664	3.644
	Highly optimal behavior	0.999	3.627	3.627	3.627	3.627	3.627

We observe that increasing the value of ϵ_r for a given value of ϵ_{ML} reduces the cost of a traveler regardless of the value of time of the traveler. This is reasonable, as increasing the ϵ_r value promotes a traveler to make optimal decision and thus reduces the costs.

In contrast, the variation of costs due to increasing value of ϵ_{ML} depends both on the value of ϵ_r and the value of time α. For example, for $\epsilon_r = 0.3$ and $\alpha = \$20/\text{hr}$, the expected cost has a peak at $\epsilon_{\text{ML}} = 0.7$, while for $\alpha = \$45/\text{hr}$, the peak occurs at $\epsilon_{\text{ML}} = 0.001$. This is because a traveler with low value of time and a high preference for GPL is better off choosing the GPL because the ML has higher associated costs. Similarly, a traveler with a high VOT and a high preference for ML is better off choosing the ML. Thus, for modeling irrational driver behavior, a lower (higher) value of ϵ_{ML} is recommended for drivers with low (high) VOT values. Though this relative setting for ϵ_{ML} values is obvious, the `OSP` model can help determine the appropriate value of ϵ_{ML}.

Figure 2.7 shows a comparison of expected costs from the `Logit` model with varying values of ζ where the x-axis is a logarithmic scale. The dashed line represents the minimum cost obtained for a certain ζ_{min} parameter. For \$20/hr, $\zeta_{\text{min}} = 0.01$ with a cost of 2.65 while for \$45/hr, $\zeta_{\text{min}} = 0.003$ with a cost of 4.08. We observe that increasing the ζ values beyond 0.1 leads to an increase in the cost, indicating that the calibrated ζ value should be lower for the LBJ network. Optimal calibration of the ζ parameter should be done, accounting for variation of VOT values in the entire population.

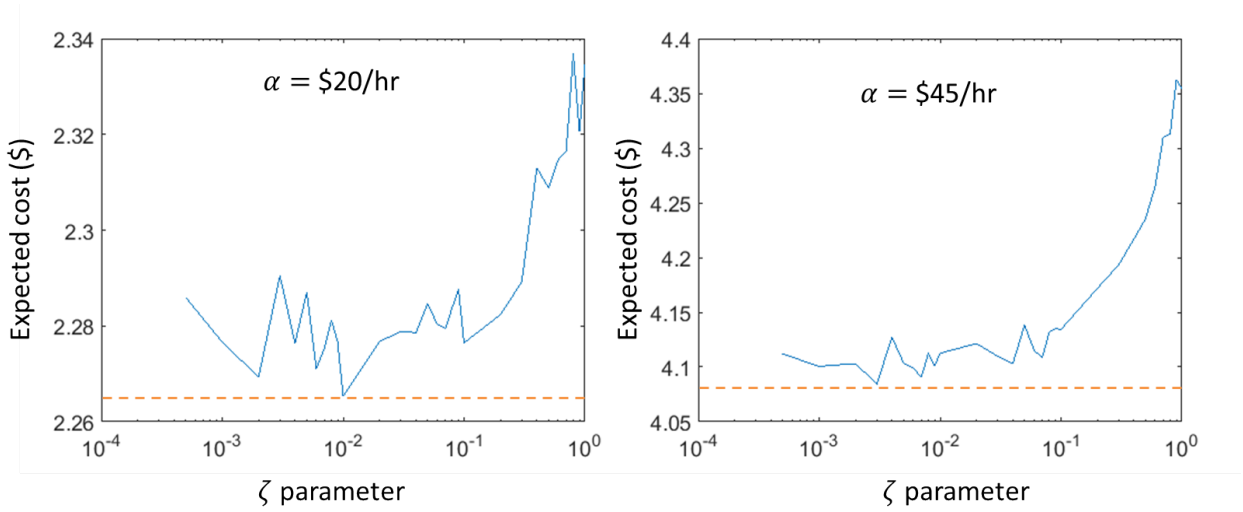


Figure 2.7: Logit model expected cost variation with ζ

Figure 2.8 shows the contour plots of the expected cost from the OSP model for varying values of ϵ_r and ϵ_{ML}. We show the approximate contour lines corresponding to ζ_{min} for both VOT values on the plot. For the case of \$20/hr, the contour line is a single dot in the corner.

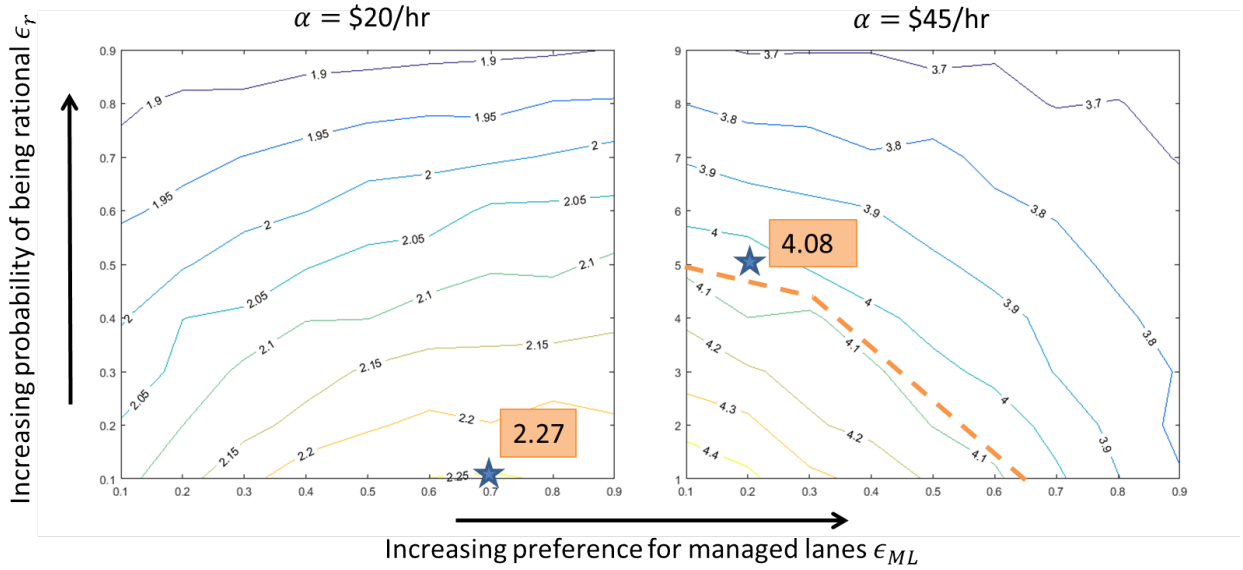


Figure 2.8: Contour plots of expected cost for varying ϵ_r and ϵ_{ML} values with approximate contour lines shown for the Logit model costs

As observed, for a given value of ζ, similar costs can be obtained from the OSP model for different combinations of ϵ_r and ϵ_{ML}. This shows that Logit model does predict "optimal" costs under irrational traveler behavior assumptions for certain values of ζ; however,

calibration for the appropriate value of ζ may be a cumbersome task. Using field-collected data on route choice to calibrate the values of ζ, ϵ_r, and ϵ_{ML} is left for future work.

2.4 Summary

In this chapter, we evaluated the performance of route choice models on STV managed lane networks with multiple entrances and exits. We formulated the online routing problem as an MDP and used the backward recursion algorithm to determine the optimal policy. The performance of other algorithms in the literature was evaluated against the online algorithm for different assumptions on driver behavior.

Simulation results on three test networks showed that the `Decision Route` model performs the best and generates an average percent error of 0.93% in the expected cost.For the tests using the `Offline` model, the peak value of online information was found to be 38 and 85 seconds for the LBJ and 13En14Ex network, respectively, which shows that providing real-time information can reduce traveler's expected cost by an amount that varies with networks size. The `Logit` model showed an average 50% error in the expected cost under the assumption of rational driver behavior; however, it obtained costs similar to the `OSP` model for certain parameter values under irrational driver behavior assumptions.

We recommend the use of `Decision Route` model for route choice under real-time information settings as it relies only on instantaneous costs, is easy to implement, and predicts close-to-optimal behavior. `Logit` model is recommended for modeling irrational driver behavior; however, appropriate calibration of the scaling parameter is essential. The `OSP` model is recommended to understand lane choice variations in a population after calibrating the values of ϵ_r and ϵ_{ML} using field data. These recommendations can be applied for route choice models used for several applications like, improving revenue forecasts for managed lane planning, calibrating parameters of route choice based on real-time data with heterogeneous drivers, equilibrium behavior analysis for travelers on managed lane networks, and online route guidance using navigation applications.

Chapter 3

Dynamic Pricing Model for Managed Lanes with Multiple Entrances and Exits

[1]Dynamic pricing for managed lanes with a single entrance and exit has been studied extensively [26, 27]. However, pricing managed lane networks with multiple entrances and exits poses different challenges, due to multiple decision points for travelers and varying toll rates for different segments of the network. Yang at al. [11] and Zhu and Ukkusuri [10] have used algorithms from stochastic dynamic programming and reinforcement learning literature to determine dynamic prices on network with multiple entrances and exits. However, the proposed methods work on network with simplified assumptions, for example, that travelers from managed lane cannot join back the general-purpose lane or that travel time on general purpose lane is independent of the flow diverting from the lane to the HOT lane.

This chapter is divided into two major sections, one where the same toll rate is applied across all toll gantry entrance locations (called centralized single-toll-variable dynamic pricing) and the other where different toll gantries are controlled by different agents who coordinate their actions to optimize the tolls for the system (called distributed dynamic pricing).

The primary contributions from the first section of this chapter are two fold. First, we develop a model where travelers make online decisions at each diverge point considering all routes on a managed lane network. The proposed method compares utility across a set of routes which grow quadratic with network size at each diverge point. Such online route choice on managed lane networks has not been studied in the literature to the best of our knowledge.

[1]The chapter has been published as following:
a) Venktesh Pandey and Stephen D Boyles. Dynamic pricing for managed lanes with multiple entrances and exits. *Transportation Research Part C: Emerging Technologies*, 96:304–320, 2018; and
b) Venktesh Pandey and Stephen D Boyles. Multiagent reinforcement learning algorithm for distributed dynamic pricing of managed lanes. *In 2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2346-2351. IEEE, 2018.
The contributions of Venktesh Pandey include study conception and design, conducting simulations, data collection, analysis and interpretation of results, and manuscript preparation.

Second, we demonstrate how the value function approximation (VFA) algorithm from the approximate dynamic programming (ADP) literature can determine tolls which perform better than "greedy" tolling schemes and other heuristics used in practice. The chapter also compares the performance of different VFA initializations in convergence towards optimal toll. The results are developed for the objectives of revenue maximization and total system travel time (TSTT) minimization. The primary contribution from the second section, which builds on the findings in the first section, are the following: First, we develop a distributed model for dynamic pricing of managed lanes with multiple entrances and exits that scales for networks with multiple toll segments. And last, with an appropriate selection of the lane choice model, we develop a method to explore continuous toll action space for all agents which obviates the need to generate and evaluate discrete toll values like previously done in [10] and [21]

3.1 Literature review

The models for dynamic pricing of managed lanes in the literature involve three sub models [26]. The *lane choice model* determines how a traveler makes the choice between the managed lane and the parallel general purpose (GP) lane. The *traffic flow model* determines how traffic propagates before and after the choice of a lane is made. The *toll pricing model* determines how tolls are updated with time using a particular objective function set by the toll operator.

There are two broad categories to model lane choice in the literature. A binary logit model selects a choice which maximizes the expected utility, where the utility for each alternative (managed lane or GP lane) is defined in terms of travel time and toll and is assumed to have a random component having a Gumbel distribution. The second approach uses a value of time (VOT) distribution directly, assuming no inherent randomness in the decision of travelers and that travelers make different choices based on their value of time. Gardner et al. [26] highlight results where the logit model gives counter-intuitive results in cases of low congestion and indicate better performance of the VOT distribution based method. The analysis in this chapter thus uses VOT distribution to model lane choice.

The traffic flow models can be broadly categorized into microscopic, mesoscopic,

and macroscopic models. Several researchers have used detailed microsimulation models to capture the precise impact of lane change around the decision point [28, 37]; however, the flexibility and faster computation time for mesoscopic models make them more useful. The most commonly used mesoscopic model is the cell transmission model used in several managed lane studies [10, 11, 27, 38, 39]. The choice of toll-pricing model is varied across the literature and depends on the operator's objective, some of which include maximizing revenue [11, 40], minimizing total system travel time [10], and maximizing corridor throughput [1].

3.1.1 Single control variable

A managed lane network with single entrance and single exit has been studied extensively in the literature. Gardner et al. [26] used value-of-time based toll pricing model for a single entrance-exit managed lane network with a downstream bottleneck on the GP lane. An extension to the same research considered stochastic demand and compared the performance of four pricing schemes [41]. Lou et al. [27] developed a model using real-time loop detector measurements to estimate logit model parameters and the optimal toll price simultaneously. Toledo et al. [38] developed a model where the real-time measurements are used to predict the toll profile in the prediction horizon and based on driver's response to the toll profile, the toll is updated in the next time step. Goccmen et al. [40] presented a revenue maximizing toll pricing model and compared adaptive and non-adaptive policies used for these purposes. They indicate that optimal revenue maximizing policies follow a "jam-and harvest" approach, where the tolls are set high in the beginning to push the GP lane towards congestion, after which the revenue on the managed lane is maximized.

Analysis of managed lane networks with multiple entrances and exits is less developed. Michalaka et al. [42] developed simulations to evaluate the effectiveness of four tolling strategies on I-95 express lanes in Florida which have multiple entrances and exits. However, the toll pricing is assumed constant and is not optimized. Dorogush and Kurzhanskiy [43] proposed a theoretical model where the optimal split of travelers at each diverge point is calculated first and then toll prices are set to achieve that proportion. Yang et al. [11] developed a distance-based revenue maximization problem for managed lane networks with multiple access points. Using additional assumptions, the problem is reduced to a stochastic dynamic

programming problem and solved on a network with three entrances and exits. However, the assumptions made are non-trivial. These include the assumption that travelers from the managed lane do not re-enter the GPL and that the travel time on the GPL is unaffected by the shift of travelers to managed lanes. Zhu and Ukkusuri [10] solve a similar problem in a connected vehicle environment where the travelers can choose the managed lane at the end of each time step; however, it is assumed that the travelers do not exit back to the GPL. They use a reinforcement learning algorithm to solve the formulated infinite horizon Markov decision process (MDP) and present the results of their analysis on the Sioux Falls network. Tan and Gao [12] present a hybrid method based on model predictive control which optimizes the toll rate between each entrance and exit to maintain the desired density on the ML. The model determines the inflows at each toll entry as a control and provides a one-to-one mapping between the inflows and the toll rate which is made possible because of simplified lane-choice assumption where a traveler do not exit the ML if they enter the lane at the current toll entry point. This chapter relaxes these assumptions by considering route choice on managed lane networks where a traveler can enter or exit the managed lane at any diverge point.

The work in Zhu and Ukkusuri [10] indicate the usefulness of ADP algorithms to solve decision making problems in transportation networks. Researchers in the field of active traffic management (ATM) strategies have used methods from stochastic optimal control to determine optimal control algorithms. Kotsialos et al. [44] developed a discrete time problem for solving coordinated and integral control of motorway networks using ramp metering and variable speed limits as ATM strategies. They use feasible-direction algorithm, originally proposed in Papageorgiou and Marinaki [45], to solve the non-linear optimal control problem operating in the policy space. Several other researchers have looked at individual ATM strategies and developed optimal control algorithms including model predictive control based algorithm in Hegyi et al. [46] and feedback linearization algorithm in Zhang and Ioannou [47]. However, these algorithms are challenging to directly apply to the managed lane pricing problem because of the level of service constraint on the managed lane and the dynamic decisions made by the travelers at each entrance point. We propose to solve the managed lane pricing problem by formulating it as an MDP, the details for which are provided in the

next section.

3.1.2 Multiple control variables

Reinforcement learning (RL) algorithms have been used for optimal decision making in several traffic control problems involving active traffic management. Mannion et al. [48] provide a review of the RL algorithms applied to the adaptive traffic signal control problem where traffic signal timings are adjusted in real-time to optimize system performance. Other researchers have looked at RL algorithms for ramp metering [49, 50] and variable speed limits [51] among other active traffic management strategies.

Applied in a multiagent setting, MARL algorithms use the MDP architecture to determine coordinated actions across all agents. Rezaee [49] and El-Tantawy et al. [52] present a coordination graph based approach to determine optimal coordinated action for multiple agents for the ramp metering and traffic signal control applications, respectively. Kuyer et al. [53] use a max-plus algorithm for coordination of signals, which is built on the coordination graph concept. The neighboring agents in a max-plus algorithm negotiate and choose optimal actions after the negotiation. However, the negotiation process can be time consuming. When applied to traffic settings, the MARL problem is assumed to have the following characteristics: full cooperation between different agents; fix locations of the agents in space; and fixed shared objective across all agents (like minimize total delay or maximize revenue etc.) [49]. This architecture simplifies the challenges of MARL like goal specification, and trade off between stability and adaptability which occur frequently in other application domains of MARL (discussed in detail in Busoniu et al. [54]). In this chapter, we build on a sparse cooperative Q-learning approach proposed in Kok and Vlassis [55].

3.2 Centralized single-toll-variable dynamic pricing

3.2.1 Optimization model

This section explains the model details for managed lane networks with multiple entrances and exits and their characteristics.

Assumptions

The assumptions made in the model are:

1. Travelers have information about the current travel time and toll information at every instant when a lane choice is possible. This assumption is reasonable in a connected vehicle environment or using variable message signs (VMSs).
2. Travelers make lane choice decisions *en route* at each diverge point based on the latest information presented. The decisions are assumed to be based on the perceived utility of the alternative at the current instant of time, without considering the choices or information that will be made available in the future. We assume that drivers rely only on the real-time information displayed on the VMSs for their decisions. Future work will extend the lane choice model for travelers who chose routes based on their past experiences.
3. There is only one origin and destination point for all travelers in the network. This assumption can be relaxed by including additional entrance and exit points and by disaggregating the traffic flow based on its destination.
4. Travelers pay the toll rate they see while entering the managed lane and continue to pay that rate until the next decision point.
5. We use the VOT distribution method discussed in Gardner et al. [26] for modeling lane choice. A discrete VOT distribution is used with different classes of vehicles and is assumed to be known beforehand. Such a distribution can be estimated by looking at historical trends of lane choice behavior of travelers, or by using estimation models like the ones discussed in Pandey [56].
6. We assume that the toll prices are optimized for a finite time horizon. The demand distribution is assumed to be known for this horizon and is considered deterministic.

In our model, we represent a general managed lane network as a combination of nodes and links. Figure 3.1(a) shows one such network. The upper set of links represent managed lanes while the lower set of links represent GPLs. Travelers make decisions at each diverge point. For the network in Figure 3.1(a), the diverge points are $1, 3, 5$ and 7. The origin and destination nodes are marked as O and D respectively.

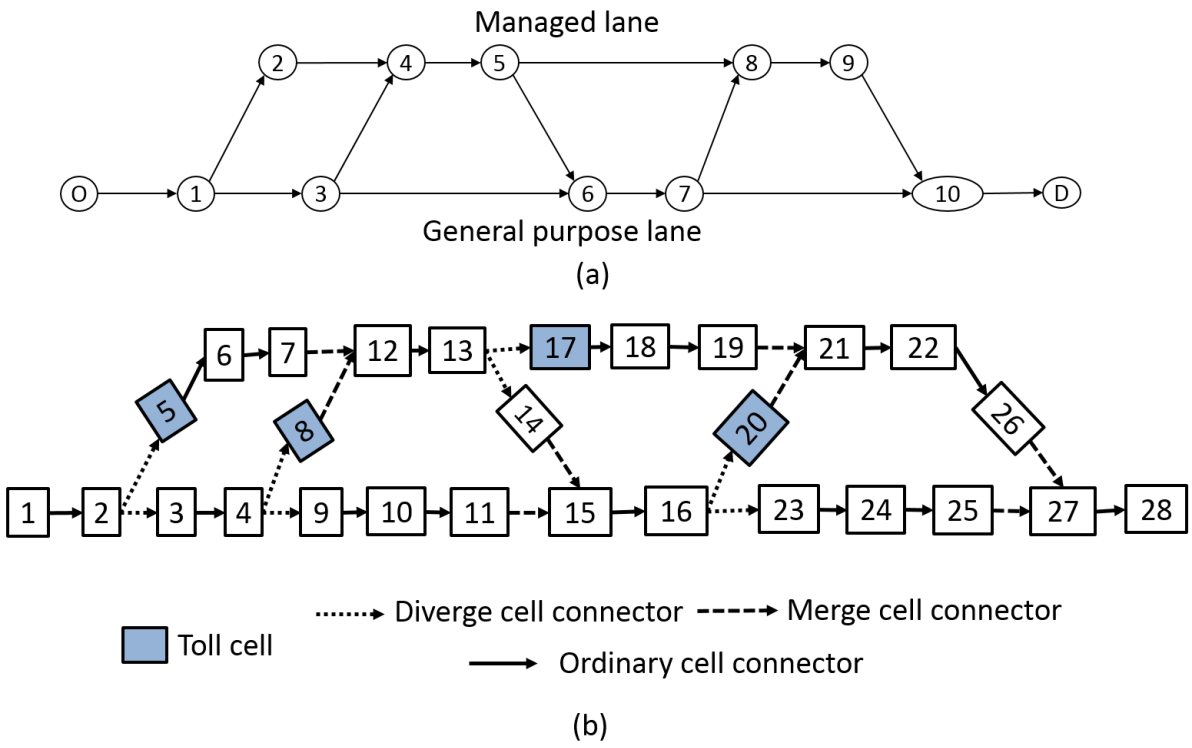


Figure 3.1: (a) Multiple entrance multiple exit managed lane network (b) Representation of the same network in cell transmission model

Notation

This section introduces the notations based on the cell transmission model [57] that is used to model traffic flow.

The time horizon is divided into equal time steps, each Δt units long. We assume $\Delta t = 1$ without loss of generality. The set of all time intervals is given by $\mathscr{T}$. Each link in the network is divided into cells of length Δx_i, where i is the cell index. The length of the cell is equal to the product of the free-flow speed in the cell, defined as ν, and the length of a time interval ($\Delta x_i = \nu \Delta t$) [57].

Set $\mathscr{C}$ represents the set of all cells in the networks. This set can be partitioned into the sets of ordinary cells $\mathscr{C}_O$, diverging cells $\mathscr{C}_D$, merging cells $\mathscr{C}_M$, source cells $\mathscr{C}_R$, and sink cells $\mathscr{C}_S$. The definition of different cell types can be found in Daganzo [57]. We define a toll cell as the cell immediately after the diverge point which leads a traveler towards the managed lane. The toll is charged only for vehicles entering this cell. Set $\mathscr{C}_{\text{toll}}$ represents the set of toll cells. Additionally, we denote the set of all cells on managed lane as $\mathscr{C}_{\text{ML}}$ and the set of all cells

on general purpose lane as $\mathscr{C}_{\text{GPL}}$, where $\mathscr{C}_{\text{GPL}} = \mathscr{C} \setminus \mathscr{C}_{\text{ML}}$. For the network shown in Figure 3.1(b), each of these sets are given as follows: $\mathscr{C}_D = \{2, 4, 13, 16\}$, $\mathscr{C}_M = \{12, 15, 21, 27\}$, $\mathscr{C}_R = \{1\}$, $\mathscr{C}_S = \{28\}$, $\mathscr{C}_O = \{3, 5, 6, 7, 8, 9, 10, 11, 14, 17, 18, 19, 20, 22, 23, 24, 25, 26\}$, $\mathscr{C}_{\text{toll}} = \{5, 8, 17, 20\}$, and $\mathscr{C}_{\text{ML}} = \{6, 7, 12, 13, 17, 18, 19, 21, 22\}$.

A cell i is connected to cell j by a cell connector (i, j). The set of cell connectors is given by $\mathscr{E}$. These connectors can be further classified as connectors going into a merge cell $\mathscr{E}_M$, connectors going out of a diverge cell $\mathscr{E}_D$, and all other cell connectors $\mathscr{E}_O = \mathscr{E} \setminus (\mathscr{E}_M \cup \mathscr{E}_D)$. Sets $\Gamma(i)$ and $\Gamma^{-1}(i)$ refer to the set of successor and predecessor cells for a cell $i \in \mathscr{C}$. Figure 3.1(b) shows the location of these cell connectors for the example network shown. Additionally, a sequence of adjacent cells defines a route, denoted by π.

A trapezoidal fundamental diagram is used to model traffic with the following parameters: free-flow speed ν, backwave speed w, capacity q_c, and jam density k_j. We assume the fundamental diagram to be uniform across the network, but the assumption can be easily relaxed. Converting these parameters to the level of a cell, we determine $N_i = k_j \Delta x_i$ as the maximum number of vehicles that can be stored in cell i and $Q = q_c \Delta t$ as the maximum number of vehicles that can leave or enter cell i in one time step.

The discrete VOT distribution used to model lane choice is given by a set of possible VOT values $v \in V$. We define $x_i^v(t)$ as the number of vehicles of VOT class v in cell i at time t and $y_{ij}^v(t)$ as the number of vehicles of class v moving from cell i to cell j from time step t to $t + 1$. The total number of vehicles $x_i(t)$ and the total flow $y_{ij}(t)$ are obtained by adding the respective variables for each VOT class. Variable $d_i(t)$ is defined as the demand entering origin cell $i \in \mathscr{C}_O$ from time t to $t + 1$.

For the distance-based pricing model, tolls are updated periodically after a certain interval. Let $K \subset \mathscr{T}$ denote the set of time steps $t \in \mathscr{T}$ at which the tolls prices are updated. We index the elements of this set by $k \in K$ and refer these as toll-update time steps. We assume that the toll-update time steps are uniformly spaced with a gap of Δk time steps. Let $\beta(k)$ denote the toll charged for all time steps between time step k and $k + \Delta k$. For a time step $t \in \mathscr{T}$, we denote by t_k the toll update time step $k \in K$, such that $k \leq t < k + \Delta k$. The units of the toll is \$/km and it can take values from a finite feasible set B. Furthermore, for each cell $i \in \mathscr{C}_{\text{toll}}$, we define l_i as the length of travel on the managed lane for which a

toll needs to be paid while entering the managed lane through cell i.

Lane choice at diverge cells

At each diverge point, a traveler of VOT class $v \in V$ compares the utility across different routes which we define as decision routes. In the previous literature, two decision routes are considered at each diverge point, defined by the set of links on managed lane and GPL until the destination point. However, this definition assumes that a traveler entering the managed lane will not choose to exit it until the destination point is reached.

To relax this assumption, on the other extreme, we could compare utilities across all possible routes towards the destination at each diverge point. We call this approach the complete-route generation (CRG) approach to determining decision routes. Figure 3.2(a) shows the routes using the CRG approach. However, this approach suffers from two disadvantages. First, the total number of such routes can grow exponentially with network size. Second, this method generates longer routes and using instantaneous time to predict utility of longer routes increases the error in the estimate of experienced travel time on a route. (We note that this CRG approach still assumes that travelers make decisions online at each diverge point.) We overcome these disadvantages by defining decision routes at each diverge cell, which compares utility across shorter routes and tractably enumerates the routes at each diverge. This new approach termed sub-route generation (SRG) approach to determining decision routes is explained next.

In the SRG approach, the set of decision routes at each diverge point comprises of the routes connecting current diverge point with the point where the next exit from the managed lane will merge the GPL. First, we number all cells in topological order, such that for each cell connector $(i, j) \in \mathscr{E}$, j is greater than i. A topological order always exists because the traffic on the corridor flows in a particular direction, so the network is acyclic. For a diverge cell i located on GPL ($i \in \mathscr{C}_D \cap \mathscr{C}_{\text{GPL}}$), we define an end cell c_i^{end} as the merge cell located on the GPL with least topological order value among all cells with order values greater than the order of cell i. Similarly, for a diverge cell i located on managed lane ($i \in \mathscr{C}_D \cap \mathscr{C}_{\text{ML}}$), we define an end cell c_i^{end} as the merge cell located on the GPL with second least topological order value among all cells with order values greater than the order of cell i. In simpler terms, the end cell is the first merge cell on the GPL located immediately after the first exit

from the managed lane beyond the current diverge point. For the network shown in Figure 3.1(b), the end cells are $c_2^{\text{end}} = 15, c_4^{\text{end}} = 15, c_{13}^{\text{end}} = 27$, and $c_{16}^{\text{end}} = 27$.

Then for each diverge cell $i \in \mathscr{C}_D$, the set of decision routes Π_i is given by all paths connecting cell i and c_i^{end}. The intuition behind such a definition is to consider the set of routes at each diverge until the next managed lane exit is found, because travelers commit to paying a certain toll when they enter the managed lane and continue paying that toll until the exit where they make the decision again. Since the managed lane must exit to the GPL (at least at the end of the corridor) and the merge cell on GPL is located at the end of that exit, an end cell always exists for every diverge point. Figure 3.2(b) shows the decision routes at each diverge point for the SRG approach. We also generalize the definition of the set of decision routes for any cell other than a diverge cell as a singleton containing the route connecting the cell and the cell immediately downstream.

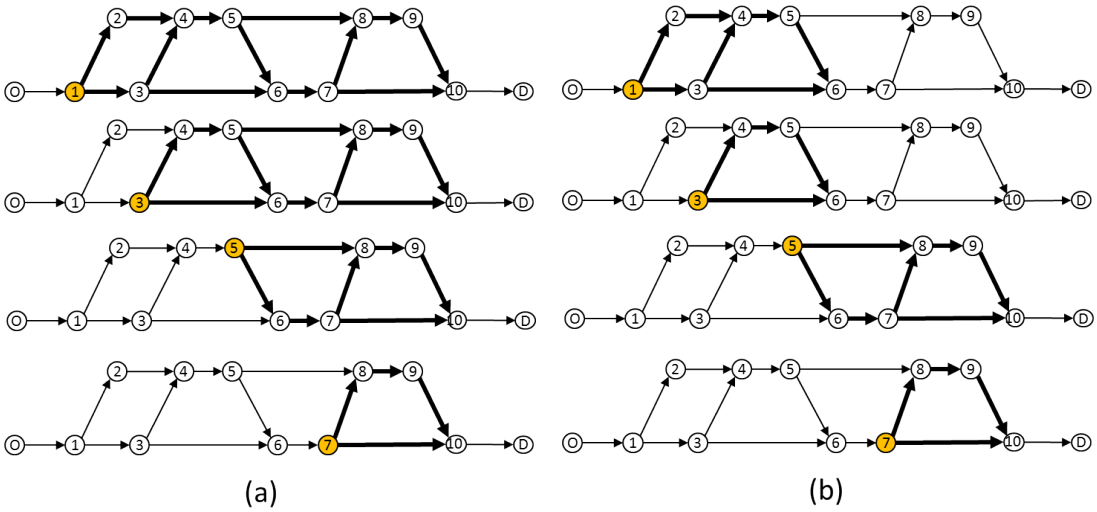


Figure 3.2: Decision routes for each highlighted diverge point using (a) the complete-route generation (CRG) approach, and (b) the sub-route generation (SRG) approach

We next show that any route connecting origin and destination in the entire network can be constructed using the decision routes in the SRG approach.

Proposition 1. *Every route in the entire network can be constructed using the decision routes in the SRG approach*

Proof. Consider a route π from source cell s_o to sink cell s_i in the entire network. Consider all diverge cells along the route and label them in topological order $(1, 2, \ldots, n)$. Since decision

routes for any cell other than a diverge cell pass through the cell immediately downstream, subroutes connecting s_o and diverge cell 1 and connecting diverge cell n and s_i can be constructed as an overlap of decision routes. Additionally, the subroute from any diverge cell i ($i < n$) to diverge cell $i+1$ can also be constructed using decision routes at cell i. There can be two cases for this construction: case 1, the decision routes from i pass through $i+1$, or case 2, the decision routes from i do not pass $i+1$. For case 1, the subroute connecting i and $i+1$ exists by default. For case 2, the decision routes at i terminate at c_i^{end}, which is a merge cell and which has a subroute connecting to the succeeding diverge cell that must be the diverge cell $i+1$ (as decision routes for any cell other than a diverge cell is a singleton connecting it to the next cell).

We can then construct π as an overlap of these sub routes constructed from the decision routes. □

The following examples show this route construction for sample routes in the network shown in Figure 3.1(a):

- For the route $O \to 1 \to 3 \to 6 \to 7 \to 8 \to 9 \to 10 \to D$, the diverge cells along the route are $\{2, 4, 16\}$. Origin is connected to cell 2 and the decision routes for 16 terminate at cell 27 which is then connected to the destination. Cell 2 is connected to cell 4 because decision routes for cell 2 pass through cell 4 (case 1 from the proof). Cell 4 is connected to cell 16 because decision routes from cell 4 terminate at cell 15 which is then connected to cell 16 (case 2 from the proof).

- For the route $O \to 1 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 9 \to 10 \to D$, the diverge cells along the route are $\{2, 4, 13, 16\}$. Cell 4 is connected to cell 13 through decision routes of cell 4 and cell 13 is connected to cell 16 through decision routes of cell 13 (both belonging to case 1 from the proof). Other diverge cells are connected from the previous example.

The number of decision routes at each diverge point contributes to the computational complexity of evaluating utilities across routes. Earlier models in the literature considered two decision routes from any diverge cell to the destination (one along managed lane and

the other along GPL) which offers lower computational complexity at the expense of not including the complete route set. The CRG approach considers the complete route set but the number of routes scale exponentially with network size. We next show that the decision routes using the SRG approach grow at most quadratically with network size. In practice, where the entrances and exits are closely located, the growth is usually linear.

Proposition 2. *The number of decision routes in the SRG approach at any diverge point is no more than* $|\mathscr{C}_D|$

Proof. Consider a diverge cell i and its corresponding end cell c_i^{end} labeled in a topological order. Let K_i be the set of all diverge cells between i and c_i^{end}. If we assume for an instant that $|K_i| = 0$, then the current number of routes connecting i and c_i^{end}, that is $|\Pi_i|$, is 2. Now add a diverge cell with topological order label between i and c_i^{end} such that c_i^{end} does not change. Adding this diverge cell increases the cardinality of the set of route choices by one, changing $|\Pi_i|$ to 3. Likewise, if we keep on adding diverge cells between i and c_i^{end} without altering the end cell c_i^{end} for cell i, the cardinality of route set keeps increasing by one. Thus for a given diverge cell, the number of decision routes is less than or equal to one plus the number of "in-between" diverge cells defined as diverge cells having topological order higher than the current cell and lower than the end cell. Since in the worst case, the number of "in-between"' diverge cells can be equal to the number of diverge cells minus one, we prove the proposition. □

Now given that there are $|\mathscr{C}_D|$ diverge cells in the network and Proposition 2 holds for each of them, we can claim that the total number of decision routes in the complete network is upper bounded by $|\mathscr{C}_D|^2$, which is a quadratic function of network size.

This SRG approach for defining decision routes thus offers three advantages: consideration of all possible routes towards the destination; tractable quantification of routes at each diverge node for online decisions; and, producing shorter route segments leading to reduced error in instantaneous travel time estimates. This makes the SRG approach suitable for use in lane choice models. More complex lane choice models can also be considered like those involving the assumptions of bounded rationality or asymmetric VOT preferences, but the study of those is left for future work.

Traffic flow equations

To explain the evolution of traffic, we follow the equations of cell transmission model proposed in Daganzo [57]. The equations are discretized for each value of time class. In this section, we only include the equations for flow from a diverge cell to its successor cells. The traffic flow equations for other cells are identical to the equations in Daganzo [57], except for discretization for each VOT class. Readers are directed to that reference for more details.

To describe the traffic flow equations for a diverge cell, we define a few additional terms. The instantaneous travel time of a cell i at time t is denoted by $\tau_i(t)$. It is measured by evaluating the average speed for the given number of vehicles in the cell using the fundamental diagram and using it to determine the average time as shown in Equation (3.1).

$$\tau_i(t) = \max\left\{\frac{\Delta x_i}{\nu}, \frac{\sum_v x_i^v(t)}{Q}, \frac{\sum_v x_i^v(t)}{\frac{W}{\Delta x_i}(N_i - \sum_v x_i^v(t))}\right\} \tag{3.1}$$

We denote the utility of a route π for a vehicle of VOT class v at time t, by $U_\pi^v(t)$. It is defined as shown in Equation (3.2). The first term computes the disutility caused by adding the instantaneous travel time over all cell j contained in the route weighted by the value of time of the vehicle. The second term computes the total toll paid on the route by computing the product of current toll rate and length of travel associated with each toll cell located on the route.

$$U_\pi^v(t) = -\sum_{j\in\pi} \tau_j(t) VOT_v - \sum_{j\in\pi, j\in\mathscr{C}_{\text{toll}}} \beta(t_k) l_j \tag{3.2}$$

Then, calculations are performed to compute diverge proportion towards the successor cells for each diverge cell. Let $\pi(k)$ denote the k-th cell in the sequence on route π. Further, define cell $c_i^v(t) \in \Gamma(i)$ as the cell chosen by the vehicles of VOT class v at diverge cell i at time t. The value of $c_i^v(t)$ is evaluated using Equation (3.3). First, $\pi_{i,\max}^v(t)$ is evaluated as the route which maximizes the utility for VOT class v among all decision routes Π_i of the diverge cell i. Then, the successor cell for each VOT class is chosen by picking the second cell in that route.

$$\pi^v_{i,\max}(t) = \underset{\pi \in \Pi_i}{\operatorname{argmax}}\, U^v_\pi(t) \tag{3.3a}$$

$$c^v_i(t) = \pi^v_{i,\max}(t)(2) \tag{3.3b}$$

Then the flow from diverge cell i towards each successor cell $j \in \Gamma(i)$ at time step t is evaluated using Equation (3.4a). The minimization is performed over three terms: the first term is the number of vehicles trying to enter cell j from cell i, the second term is the capacity, and the third term is the number of vehicles allowed to enter cell j given the current number of vehicles in cell j.

$$y_{ij}(t) = \min \left\{ \sum_{v \in V, c^v_i(t)=j} x^v_i(t), Q, \frac{w}{\nu}\left(N_j - \sum_{v \in V} x^v_j(t)\right) \right\} \quad \forall j \in \Gamma(i) \tag{3.4a}$$

$$y^v_{ij}(t) = \frac{x^v_i(t)}{\sum_{v \in V, c^v_i(t)=j} x^v_i(t)} y_{ij}(t) \quad \forall v \in V \quad \forall j \in \Gamma(i) \tag{3.4b}$$

This flow from diverge cell i is then discretized for each VOT class in proportion of the current number of vehicles in that VOT class in the diverge cell trying to enter the same successor cell j, using the Equation (3.4b).

The values of $y^v_{ij}(t)$ are evaluated for every other cell using the methods in Daganzo [57] and these are then used to update the number of vehicles in each cell in the next time step using Equation (3.5) which is based on flow conservation.

$$x^v_i(t+1) = x^v_i(t) + \sum_{k \in \Gamma^{-1}(i)} y^v_{ki}(t) - \sum_{j \in \Gamma(i)} y^v_{ij}(t) \quad \forall v \in V, \forall i \in \mathscr{C} \tag{3.5}$$

Optimization problem and Markov decision process

The objective of the optimization problem is to find toll rate $\beta(k)$ for all time steps $k \in K$, such that a certain objective is achieved. Two objectives are considered in this analysis: revenue maximization and TSTT minimization. The objective function for revenue maximization is given in Equation (3.6) where the revenue collected at each toll-update time step, evaluated by charging toll on all vehicles entering each toll cell till the next toll update,

is summed over all toll-update time steps. The objective function for TSTT minimization is given in Equation (3.7) which sums over all time steps the number of vehicles present in each cell per time step.

$$Z_{\text{Rev}} = \max_{\beta(k)} \sum_{k \in K} \sum_{t=k}^{k+\Delta k} \sum_{i \in \mathscr{C}_{\text{toll}}} \left\{ \beta(k) l_i \sum_{j \in \Gamma^{-1}(i)} y_{ji}(t) \right\} \tag{3.6}$$

$$Z_{\text{TSTT}} = \min_{\beta(k)} \sum_{k \in K} \sum_{t=k}^{k+\Delta k} \sum_{i \in \mathscr{C}} x_i(t) \tag{3.7}$$

Three constraints are included in the optimization problem. The first constraint requires variables $x_i(t)$ and $y_{ki}(t)$ to be updated using the traffic flow equation of the cell transmission model. The second constraint requires the toll rate $\beta(k)$ to belong to the defined set B. The last constraint requires that the managed lane always maintains a speed above a desired minimum speed limit $\nu_{\text{min}} \leq \nu$. We model this constraint by restricting the number of vehicles allowed in the managed lane to be less than or equal to a threshold value. For a trapezoidal fundamental diagram with parameters defined above, we define ν_{thresh} as the speed at which the fundamental diagram changes from the horizontal to the downward sloping curve. This value is evaluated as $\nu_{\text{thresh}} = \frac{q_c}{k_j - q_c/w}$ and is given by the slope of the dashed line shown in Figure 3.3.

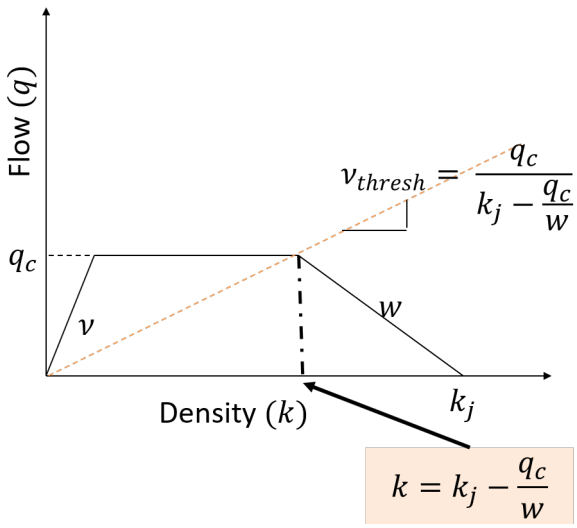


Figure 3.3: Trapezoidal fundamental diagram for modeling traffic flow

We restrict the number of vehicles allowed in the managed lane using Equation (3.8), where the maximum number of vehicles in each cell on the managed lane is determined based on the value of $\nu_{\min}$.

$$\text{if } \nu_{\text{thresh}} \leq \nu_{\min} \leq \nu, \quad x_i(t) \leq \frac{Q\Delta x_i}{\nu_{\min}} \quad \forall i \in \mathscr{C}_{\text{ML}} \tag{3.8a}$$

$$\text{else if } 0 \leq \nu_{\min} < \nu_{\text{thresh}}, \quad x_i(t) \leq \frac{wk_j\Delta x_i}{\nu_{\min} + w} \quad \forall i \in \mathscr{C}_{\text{ML}} \tag{3.8b}$$

This optimization problem is then formulated as a finite-horizon MDP. MDPs are a traditional method for solving problems that involve sequential decision making [58]. For the given optimization problem, a toll rate decision is to be made every Δk time step, so the update step of the MDP is at every toll-update time step $k \in K$. The components of the MDP are defined as the following:

- **State**: Number of vehicles of each class in each cell at the start of toll update time step k:
  $x(k) = \{(x_i^v(k))|\forall i \in \mathscr{C}, \forall v \in V\}, \forall k \in K$
- **Action**: Toll rate $u_k(x(k)) = \beta(k) \in U_k(x(k)) \subseteq B$, where $U_k(x(k))$ is the set of tolls at the current state $x(k)$ that do not violate the constraint in Equation (3.8) for all time steps t between k and $k + \Delta k$
- **Transition function**: Obtain $x(k + \Delta k)$ after applying traffic flow equations for all time steps $k \leq t < k + \Delta k$
- **One step reward**: Equation (3.9a) shows the one step reward for the revenue maximization objective, while Equation (3.9b) shows the one step reward for the TSTT minimization objective. We note that the one step reward for the TSTT minimization objective does not directly depend on the action selected in the state, but may indirectly depend on it based on the next state due to the current action.

$$g_{\text{Rev}}(x(k), u_k(x(k))) = \sum_{t=k}^{k+\Delta k} \sum_{i \in \mathscr{C}_{\text{toll}}} u_k(x(k)) l_i \sum_{j \in \Gamma^{-1}(i)} y_{ji}(t) \tag{3.9a}$$

$$g_{\text{TSTT}}(x(k), u_k(x(k)) = -1 * \sum_{t=k}^{k+\Delta k} \sum_{i \in \mathscr{C}} x_i(t) \tag{3.9b}$$

The next section describes the solution algorithm using this MDP structure to solve the dynamic pricing problem.

3.2.2 Solution methods

Solving the formulate finite-horizon deterministic MDP using traditional MDP solution methods is challenging because of the exponential increase in the number of states with the size of network. Let N be the maximum value of the number of vehicles that can be stored in any cell, $|V|$ be the number of vehicle class, and $|\mathscr{C}|$ be the number of cells, then number of states is $O(N^{|V||\mathscr{C}|})$, which grows exponentially with the number of cells in the network. We use the value function approximation (VFA) method to address this curse of dimensionality. In the following subsections, we focus our explanation on revenue maximization as the objective. The analysis for TSTT minimization will follow the same steps by replacing all max operators with a min, and using g_{TSTT} instead of g_{Rev}.

VFA is a commonly used method to solve discrete time approximate dynamic programming (ADP) problems [58]. This method attempts to determine a good estimate for the value function in each state. A value function in a state $x(k)$, indicated by variable $R^*(x(k))$, is the maximum revenue that can be obtained if the system were to start from state $x(k)$ at time k till the end of time horizon. If $R^*(x(k))$ is known accurately for each state, the optimal toll rate $u^*(x(k))$ at any given state $x(k)$ can be determined using the Bellman equation shown in Equation (3.10).

$$u^*(x(k)) = \underset{u_k(x(k)) \in U_k(x(k))}{\text{argmax}} \left(g_{\text{Rev}}(x(k), u_k(x(k))) + R^*(x(k + \Delta k))\right) \tag{3.10}$$

The VFA method initializes the value function in every state to a suitable guess and improves that value iteratively, and involves three steps [58]. The first step chooses a form for the value function. This can either be a look-up table or a parametric function. The second component simulates a particular policy and learns its values, and the last component

updates the estimates of the value functions and chooses a better policy to run the second step again. The iteration between the second and the third step is continued until convergence.

For our analysis, we chose a look-up table representation as a form for value functions as it is the easiest to implement. In this representation, the value of each state is initialized and updated independently. Such a representation works well for MDPs with a small number of states. For MDPs with a large number of states, the states are aggregated together based on their similarity. All states classified within the same aggregate are assumed to have same value for the value functions. For our analysis, we aggregate all states by rounding the current flow values in each cell with respect to each VOT class to the nearest integer. That is, all states $x(k)$ which round to the same integer value of the current vehicles in each cell $x_i^v(k)$, for all $i \in \mathscr{C}$ and $v \in V$, are assumed to have same value function at any given toll-update time step k.

After defining an appropriate form for the value function, we run the second step of VFA. A policy is selected from the current estimates of value function using the same formula in Equation (3.10). This selection procedure assumes that the current value function approximations are optimal and is referred as greedy policy selection in the literature [59]. Other policy update methods also exist, but we choose this method for its simplicity.

For the last step of VFA, we update the previous estimate by combining it with the estimate of values predicted from simulating the policy in the previous step using a stepsize parameter. The above three steps are summarized in Algorithm 3. The variable $R_n^*(x(k))$ is the estimate of the value function in state $x(k)$ at time k for the n-th iteration. $x_n(k)$ represents the state chosen at time step k in the n-th iteration. λ_n represents the step-size to combine the new and old value estimates in iteration n. We terminate the algorithm if either the maximum number of iterations is reached or the value functions do not change in the last 10 consecutive iterations.

A proper selection of step-size is crucial for convergence [58]. For our analysis, the step-size value was chosen to decrease monotonically over iterations. If $\overline{N}$ is the maximum number of iterations simulated, then the step-size at iteration n was chosen as, $\lambda_n = \frac{\overline{N}}{\overline{N}+\frac{n}{10}}$. Such a step-size selection ensures that λ_n drops gradually from 1 to 0.91.

For the purposes of our study, we used two value function initializations for the

Algorithm 3 VFA algorithm using look-up table representation

Step 0**:** Initialization
Set $R_0^*(x(k))$ to a suitable value $\forall x(k) \quad \forall k \in K$
Choose initial state $x_1(0)$
Set $n = 1$

Step 1**:** Simulating a policy
Set $\widehat{R}_n(x(k))$ =null $\forall x(k)$
for $k \in K$ **do**
$\widehat{R}_n(x_n(k)) = \max_{u_k(x_n(k)) \in U_k(x_n(k))} \left(g_{\text{Rev}}(x(k), u_k(x_n(k))) + R_{n-1}^*(x(k+\Delta k))\right)$
$u_n(k) = \operatorname{argmax}_{u_k(x_n(k)) \in U_k(x_n(k)} \left(g_{\text{Rev}}(x(k), u_k(x_n(k))) + R_{n-1}^*(x_n(k+\Delta k))\right)$
Determine $x_n(k+\Delta k)$ after applying traffic flow equations on $x_n(k)$ using toll value $u_n(k)$ for all time steps t such that $k \leq t < k + \Delta k$
end for

Step 2**:** Update the state values
for $kinK$ **do**
if $\widehat{R}_n(x(k))$ is NOT null **then**
$R_n^*(x(k)) = \lambda_n \widehat{R}_n(x(k)) + (1 - \lambda_n) R_{n-1}^*(x(k))$
end if
end for
if $n <$ max number of iterations OR $R_n^*(x(k))$ values converged **then**
$n \leftarrow n + 1$
Go back to Step 1
else
Stop. Report $R_n^*(x(t))$ as final value estimate for each state $x(t)$ in time step t
end if

revenue maximization objective:

1. **VFA1**: Initialize values for all states in time step k as $L(n(\mathscr{T}) - k - 1)$, where L is an upper bound on the one-step revenue that can be obtained from any state.
2. **VFA2**: Initialize value function in each state to be equal to the sum of the number of vehicles on general purpose lanes after each diverge point, summed over all diverge points in the network. We use a scaling factor S ($S \geq 1$) with values greater than or equal to 1 to control the relative values of initial value functions.

The VFA1 initialization uses the knowledge that a state which is farther away from the last time horizon can generate higher revenue (and equivalently has higher value) than the state at the later time step. The VFA2 initialization uses the intuition that a state with congested

GPLs allows more travelers to shift to the managed lane and thus would generate higher revenue. For the TSTT minimization objective, we initialized all state values to 0 (**VFA3**) using the simplest initialization method.

The convergence of the VFA method is not guaranteed for a general initialization. However, if the MDP is deterministic and the state and action spaces are finite, "optimistic" initializations have been proved to converge to optimal [58]. An optimistic initialization for the revenue-maximization (TSTT-minimization) objective initializes values to a very high (low) number which encourages the VFA method to explore all possible states and determine the optimal toll profile. Due to the aggregation scheme employed for the representation of states, we cannot guarantee convergence of Algorithm 3 even though we have a large but finite space for states and actions. Even if the convergence happens, it is not guaranteed to converge to optimal. Thus, Algorithm 3 serves as a heuristic to find the optimal toll profiles. We track the best-found toll profile for each iteration and use the objective from the best-found toll profile at termination as a measure of performance of this heuristic.

We compared the performance of the VFA method against three other heuristics as follows:

1. **Density-based heuristic:** This heuristic uses feedback control to keep the number of vehicles on managed lane to a desired number. The tolls are updated using Equation (3.11), where $X_{\text{HOT}}^{\text{desired}}$ is the desired number of vehicles on HOT lane, X_{HOT} is the current number of vehicles on the HOT lane, and P is the regulator parameter. The value of P is varied between 0.1 and 1.5 (in the increments of 0.1) for selecting the best tolls using this heuristic for a given objective. We refer this heuristic as `Density` in the remaining text.

$$\beta(k+1) = \beta(k) + P * (X_{\text{HOT}} - X_{\text{HOT}}^{\text{desired}}) \tag{3.11}$$

2. **Ratio-based heuristic:** This heuristic aims to keep the ratio of number of travelers on the ML to GPL near a desired value. The tolls are updated using Equation (3.12), where $r_{\text{HOT:GP}}^{\text{desired}}$ is the desired ratio between managed and GPL, $r_{\text{HOT:GP}}$ is the current ratio between managed and GPL, and P is the regulator parameter. Same as before,

the value of P is varied between 0.1 and 1.5 (in the increments of 0.1), and the value of $r^{\text{desired}}_{\text{HOT:GP}}$ is varied between 0.1 and 0.4 (in the increments of 0.05) for selecting best tolls using this heuristic for a given objective. We refer this heuristic as `Ratio` in the remaining text.

$$\beta(k+1) = \beta(k) + P * (r_{\text{HOT:GP}} - r^{\text{desired}}_{\text{HOT:GP}}) \tag{3.12}$$

3. **Myopic revenue policy:** This policy acts in a greedy fashion. At each time step, it selects the toll rate which maximizes the one-step revenue obtained all over feasible tolls. We refer this heuristic as `Myopic` in the remaining text.

The `Density` and `Ratio` heuristics are forms of feedback-control method for determining toll rate based on the current congestion pattern, similar to the feedback-control methods for other ATM strategies like ALINEA for ramp metering [60]. Though the details of dynamic pricing heuristics currently used in practice are proprietary and undisclosed, extensions of feedback-control methods are commonly used [61]. The `Myopic` heuristic is another heuristic commonly used in the MDP literature to compare the performance of the optimal policy and is thus included in our study.

3.2.3 Experiments

This section shows the results of the VFA algorithm and its comparison against other heuristics. We conduct the analysis on four test networks: a 0.5-mile long double entrance single exit (DESE) network, a 4.5-mile long managed lane corridor having a similar structure as the second toll segment on the LBJ TEXpress lanes in Dallas, Texas (LBJ), and two other artificially constructed networks, one with seven entrances and five exits (7En5Ex), and the other with thirteen entrances and fourteen exits (13En14Ex). Figure 3.4 shows schematics for the four networks. The origin and destination nodes are marked as O and D respectively. The dashed links represent the location of bottleneck in the network to simulate the effect of congestion.

The traffic flow follows a trapezoidal fundamental diagram with $\nu = 90$ km/hr, $w = 30$ km/hr, $q_c = 2200$ veh/hr/lane, and $k_j = 165$ veh/km. Each time step is assumed to be 6 seconds long and the tolls are updated every 50 time steps, that is $\Delta k = 300$ seconds. Each

network is simulated for a period of two hours with no vehicles in each cell at the beginning of the simulation. The demand distribution from origin to the destination follows the profile as shown in Figure 3.5(a). The feasible toll set B includes toll rates varying from \$0.05/km to \$1.25/km in increments of \$0.1/km. The minimum speed limit $\nu_{\min}$ on the ML is set to 80 km/hr. For the analysis we consider five VOT classes: \$10/hr, \$15/hr, \$20/hr, \$25/hr, and \$30/hr with assumed proportions of demand as $0.1, 0.4, 0.2, 0.2$, and 0.1 respectively. Figure 3.5(b) shows the discrete VOT distribution.

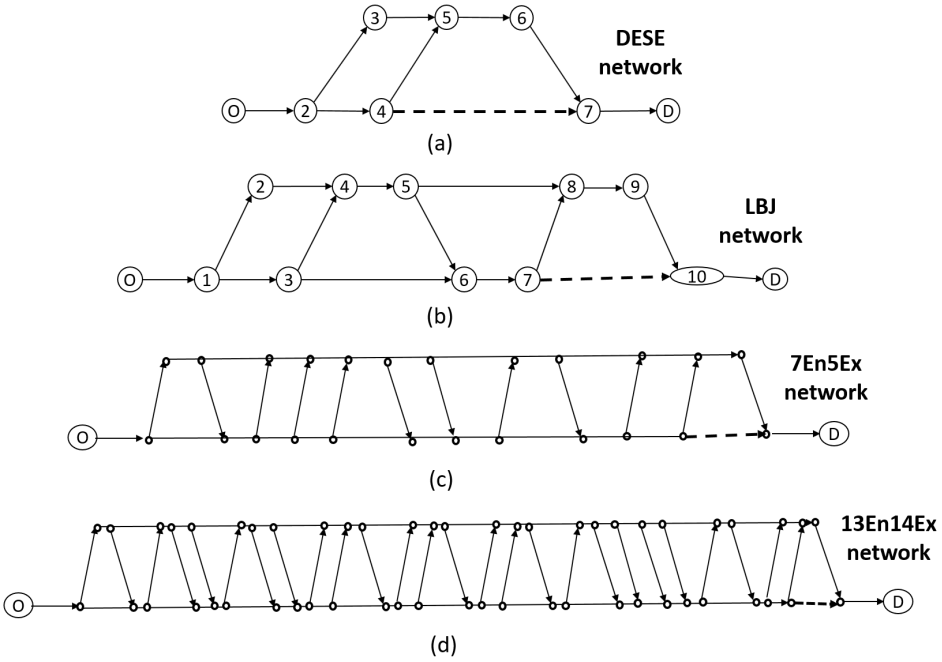


Figure 3.4: Four test networks: (a) double entrance single exit (DESE) network; (b) LBJ TEXpress network for toll segment 2; (c) a network with seven entrances and five exits (7En5Ex); and (d) a network with thirteen entrances and fourteen exits (13En14Ex). The dashed link indicates the location of downstream bottleneck.

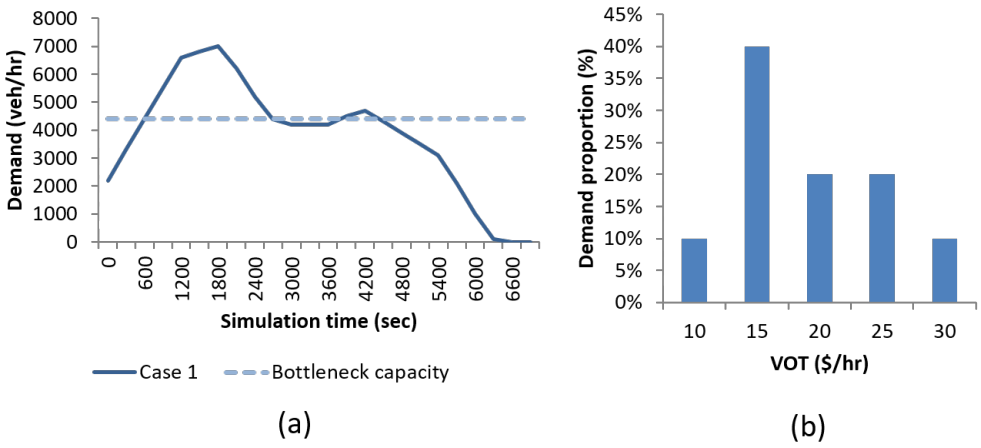


Figure 3.5: (a) Demand as a function of time (b) VOT distribution

To demonstrate the effectiveness of the SRG approach, we first compare the average computation time per iteration for solving the VFA algorithm using both SRG and CRG approaches. The algorithm is implemented in Java and the simulations are run on a 3.3 GHz Windows machine with 4 GB RAM. Figure 3.6 shows the bar plot of the average computation time per iteration for each network where the average is calculated as the total CPU time for the first 10 iterations divided by 10.

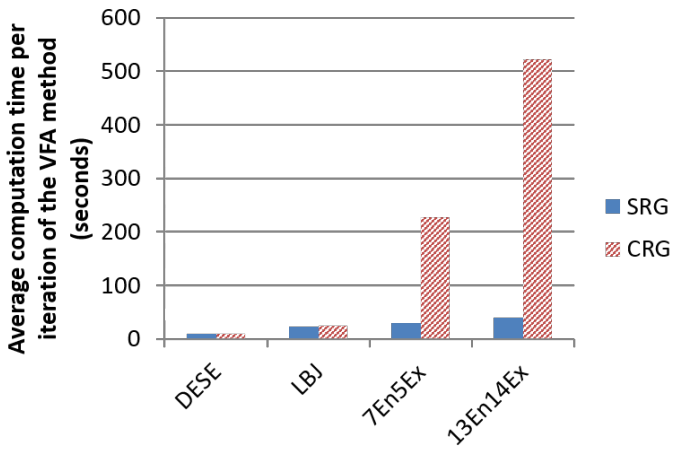


Figure 3.6: Comparison of computation time per iteration in seconds for the SRG and CRG approaches

As observed, the CRG approach requires higher computation time per iteration for all networks except DESE network where the CRG and SRG approaches perform identically because both approaches generate same set of decision routes at each diverge point. The

percent reduction in computation time using the SRG approach relative to the CRG approach is higher for networks with more entrances and exits with a reduction of 86.8% for the 7En5Ex network and 92.6% for the 13En14Ex network. This is because the CRG approach enumerates all possible combinations of decision routes which increases the computation time for determining the downstream cell at each diverge cell in the network. The results in the remaining paper use the SRG approach for the VFA method.

Next, we analyze the convergence characteristics of the VFA method for different VFA initializations for the four networks for both toll optimization objectives. The value of L for the VFA1 initialization is set as 50 while the value of S for the VFA2 initialization is set as 3.0. These values are determined based on experiments. Figure 3.7 shows the plots of variation of revenue with iterations for the revenue maximization objective for the VFA1 and VFA2 initialization. The thinner lines in the plot show the variation of revenue in each iteration obtained from simulating a toll profile using the most recent value function estimates, while the thicker lines show the variation of the maximum revenue obtained so far. The value of $\bar{N}$ is set to 1000 for the DESE and LBJ networks, and set to 50 for the 7En5Ex and 13En14Ex networks considering the constraints on computational resources.

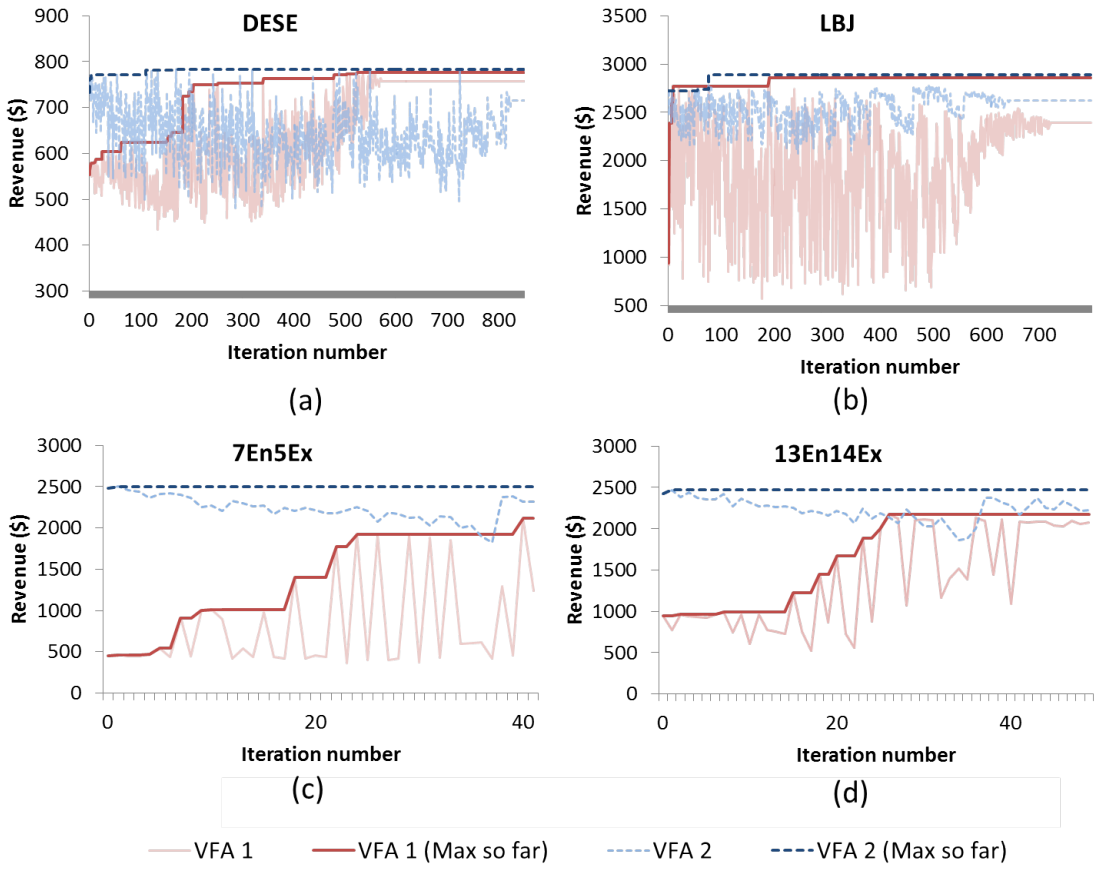


Figure 3.7: Revenue obtained as a function of iteration number for the VFA1 and VFA2 initializations for all four networks

We make two main observations from the convergence plots for the DESE and LBJ networks in Figures 3.7(a) and 3.7(b), respectively. First, both VFA initializations converge after 700-800 iterations for both networks. The oscillations indicate that the VFA method continues to explore new states; however, the method learns the values of the visited states and avoids the states which have lower values. The convergence happens when the algorithm stops learning. Second, the converged value is not optimal as it is lower than the best-found revenue obtained in the earlier iterations. The converged revenue value for the VFA1 (VFA2) initialization is 2.5% (8.7%) lower than the best-found revenue for the DESE network and 16.3% (9.38%) lower than the best-found revenue for the LBJ network. This suboptimal convergence behavior is as expected: the VFA1 and VFA2 initializations determine the value of each state relative to the other states and it is not necessary that the relative values

for each state are set correctly for the given aggregation choice.

For the 7En5Ex and the 13En14Ex networks, both initializations do not converge within 50 iterations; however, in spite of the lack of convergence, the VFA1 initialization continues to simulate toll profiles which generate high revenue as indicated by the gradual increase in the best-found revenue obtained so far.

We also observe that the maximum revenue obtained from the VFA2 initialization is higher than the one obtained from the VFA1 initialization for all four networks and that maximum is obtained at an earlier iteration than the VFA1 initialization. This is because the VFA2 initialization distinguishes between the relative values of states at a given time step by assigning higher value to states with more vehicles on GPL and thus is able to find better policies in first few iterations. This shows that VFA2 performs better than VFA1 initialization for all networks and suggests the usefulness of the VFA2 initialization for obtaining toll profiles which produce high revenue in earlier iterations.

Figure 3.8 shows the convergence plots for the VFA3 initialization for the TSTT minimization objective for the four networks. Similar to the plots in Figure 3.7, the thinner line represents the variation of TSTT obtained from the toll profile simulated in each iteration, while the thicker line represents the minimum TSTT obtained thus far.

We observe that TSTT converges for both DESE and LBJ networks, and the value at convergence is within 0.2% and 0.8% of the minimum TSTT value obtained at termination, respectively. Additionally, this minimum value is obtained earlier in the simulation (within first 5 iterations for the LBJ network and within first 50 iterations for the DESE network.) For the 7En5Ex and the 13En14Ex networks, a similar pattern is observed where the minimum TSTT after 50 iterations is obtained in the first iteration itself. This early detection of toll profiles with better TSTT is due to the nature of congestion in the network. As vehicles arrive gradually at the bottleneck, the TSTT is minimized using profiles which send more vehicles towards ML to best utilize the entire network capacity. For the VFA3 initialization, the chosen action is the one that minimizes the number of vehicles in the next state since the one step cost is same for all actions. The action minimizing the number of vehicles in the next state is the one that sends more vehicles towards the ML resulting in more vehicles entering and exiting the network. This leads to close-to-optimal behavior in the first few

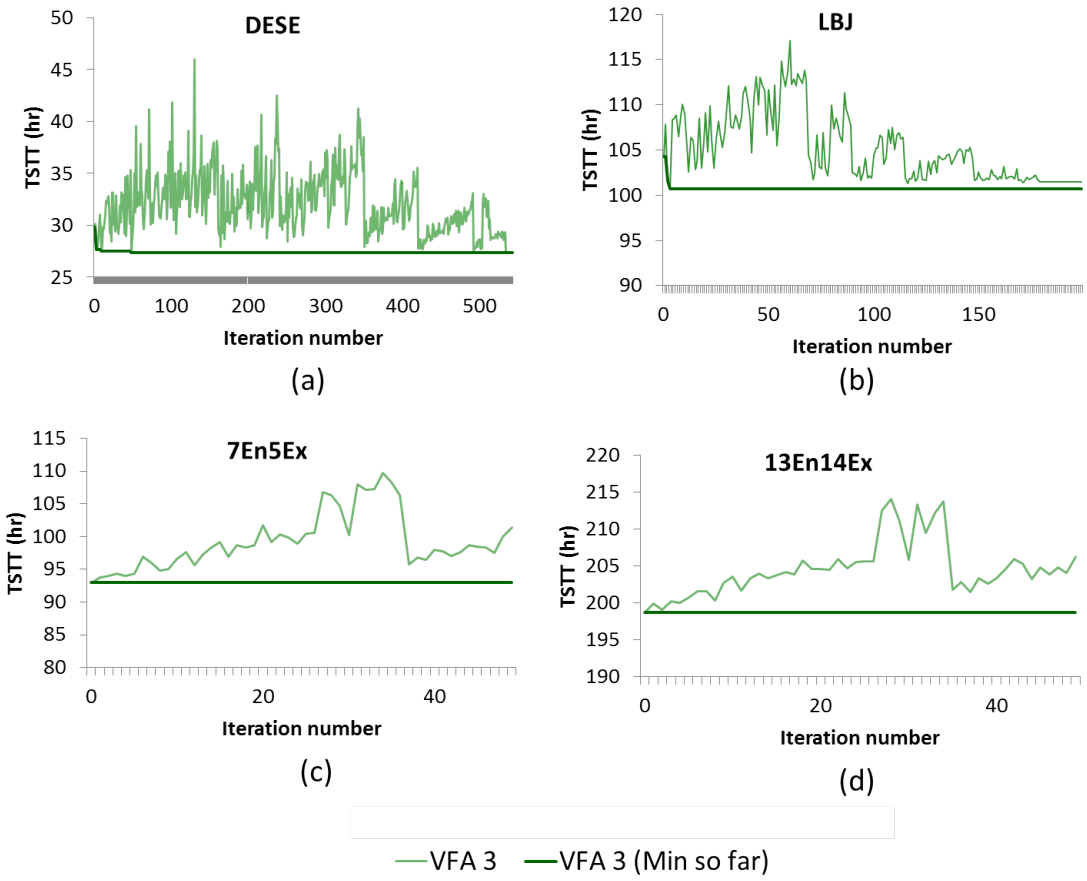


Figure 3.8: TSTT obtained as a function of iteration number for the VFA3 initialization for all four networks

iterations. Since this gradually-evolving congestion pattern is commonly observed during peak hours, VFA3 initialization is suitable for determining toll profiles with low TSTT in fewer iterations.

The convergence of VFA methods for larger networks takes more iterations which may require large computation time. However, based on the observations above, VFA method can be used as a heuristic to obtain relatively better solutions in lesser time. We recommend the use of VFA2 initialization as a heuristic for obtaining revenue-maximizing toll profiles and VFA3 initialization as a heuristic to obtain TSTT-minimizing toll profiles.

Next, we compare the toll profiles generated by the VFA method with the other heuristics. Table 3.1 shows the comparison of revenue and TSTT for the toll profiles obtained from the VFA method and the profiles generated from the heuristics for both objectives. A "best policy" from a particular VFA initialization or a heuristic is the one which leads to a

maximal return on the objective for any iteration or for any choice of input parameters of a heuristic.

Table 3.1: Comparison of revenue and TSTT for different toll policies for the four networks

Revenue maximization objective									
		DESE network		LBJ network		7En5Ex network		13En14Ex network	
		Best policy revenue ($)	Best policy TSTT (hr)	Best policy revenue ($)	Best policy TSTT (hr)	Best policy revenue ($)	Best policy TSTT (hr)	Best policy revenue ($)	Best policy TSTT (hr)
Method	VFA 1	776.93	56.24	2860.44	136.05	2120.87	143.62	2175.02	255.12
	VFA 2	783.21	85.93	2891.05	158.25	2501.73	180.34	2471.03	286.07
	Density	486.70	34.70	1535.71	126.06	257.15	99.22	138.29	203.49
	Ratio	532.45	37.94	1800.76	126.76	310.25	101.66	335.65	207.50
	Myopic	553.24	36.06	943.94	104.08	454.04	102.61	947.22	222.13
TSTT minimization objective									
		DESE network		LBJ network		7En5Ex network		13En14Ex network	
		Best policy revenue ($)	Best policy TSTT (hr)	Best policy revenue ($)	Best policy TSTT (hr)	Best policy revenue ($)	Best policy TSTT (hr)	Best policy revenue ($)	Best policy TSTT (hr)
Method	VFA 3	180.75	27.36	540.37	100.73	49.65	92.98	49.65	198.77
	Density	486.70	34.70	344.86	102.97	49.65	92.98	49.65	198.77
	Ratio	171.70	30.56	586.21	105.65	153.93	101.06	335.65	207.50

We observe that VFA3 initialization predicts a policy which has lowest TSTT across all policies. For revenue maximization, VFA2 determines the policy with maximum revenue across all other networks. Other heuristics perform poorly compared to the VFA method regardless of the choice of their parameter values. For the revenue maximization objective, the revenues generated by the `Density`, `Ratio`, and `Myopic` policies are lower than the maximum revenue obtained by the VFA method by an average of 67.2%, 60.9%, and 60.1%, respectively. This shows that the VFA method performs well for revenue maximization.

For the TSTT minimization objective, the TSTT generated by `Density` and `Ratio` heuristics are higher than the lowest TSTT obtained from the VFA method by an average of 7.3% and 7.4%, respectively. The percent differences are lower compared to the revenue maximization objective indicating that these heuristics are more suited towards the TSTT minimization objective than the revenue maximization objective.

We also observe that the policy minimizing TSTT generates a revenue which is 88.6% lower than the best-found maximum revenue on an average. Similarly, we observe that the policy maximizing the revenue generates TSTT which, on an average, is 102.3% higher

than the best-found minimum TSTT for all the networks. This indicates that both these objectives are conflicting in nature. Finding toll profiles which optimize both objectives together is part of the future work.

Figure 3.9 shows a comparison of revenues and TSTT obtained from the best policies for each method at varying levels of demand for the DESE network. The VFA method is only run for 50 iterations. As observed in Figure 3.9(a), for the demand level 0.5, all methods generate \$0 revenue as there is no congestion in the network. As the demand levels increase, the revenue from the VFA2 method increases until a threshold beyond which it starts decreasing. This is reasonable since there is an upper limit on the toll rate and thus only a certain revenue can be generated from tolling the managed lanes. The revenues from the `Density` and `Ratio` heuristics are on an average 36.8% and 40.2% lower than the revenues from VFA2 heuristic run for 50 iterations. The TSTT values in Figure 3.9(b) are almost identical for all the three methods, but the lowest TSTT is always obtained using the VFA3 initialization. The TSTT obtained from the `Density` and `Ratio` heuristics are on an average 7.8% and 4.2% higher than the TSTT from the VFA3 initialization.

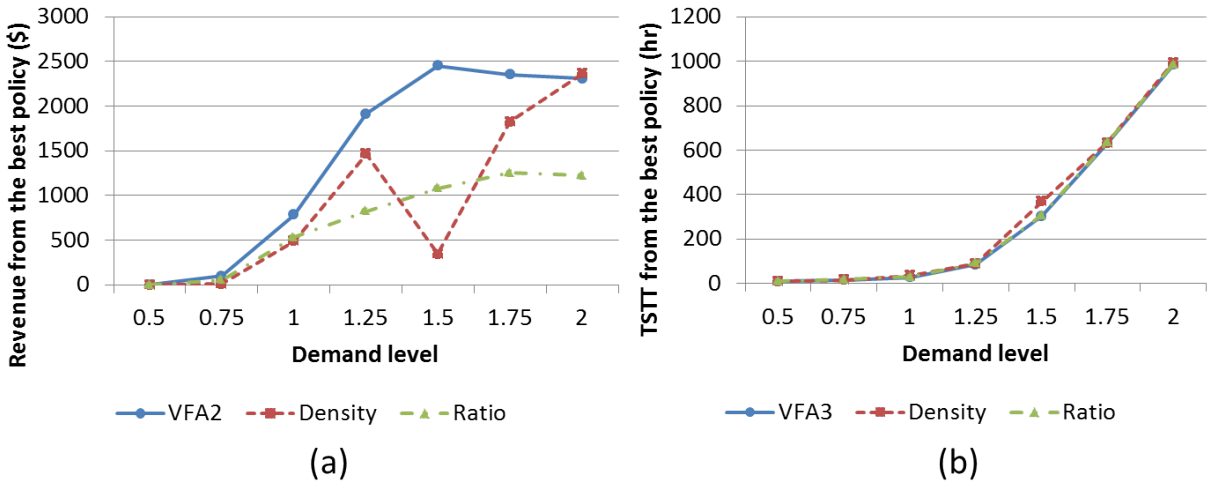


Figure 3.9: Variation of revenue and TSTT obtained from the three heuristics with varying levels of demand for the DESE network

The computation time required to run 50 iterations of the VFA method is less than 1 minute for both VFA2 and VFA3 initializations. This suggests the usefulness of VFA method for online implementation for the DESE network. The demand during the previous 5-minute

period can be observed and used to predict the future demand using a certain demand prediction algorithm. Using this demand profile, the optimal policies can be generated by running 50 iterations of the VFA method and the best toll profile predicted by the VFA method can be implemented in the field based on the optimization objective. For larger networks, where running each iteration is more time intensive, we recommend offline training of toll profiles where several demand profiles may be simulated in advance and the best course of action is made available for real-time decision based on the observed demand pattern. A detailed analysis of the effectiveness of VFA method for online implementation is left for future work.

Last, we compare the behavior of the toll profiles for different objectives. Figure 3.10 shows the variation of toll rate with time for the DESE network for the three heuristics at demand level 1 for both objectives. For the revenue maximization objective, we observe that the VFA2 profile charges toll rate between $\$1 - \$1.25$/km from 1500-2100 seconds during the simulation, which is higher compared to the rates charged by the `Density` and `Ratio` heuristics. On the other hand, for the TSTT minimization objective, the toll rates from the VFA3 profile are comparately lower than the other heuristics. The variation in the toll profile behavior for both objectives can be explained by visualizing the evolution of congestion in the network.

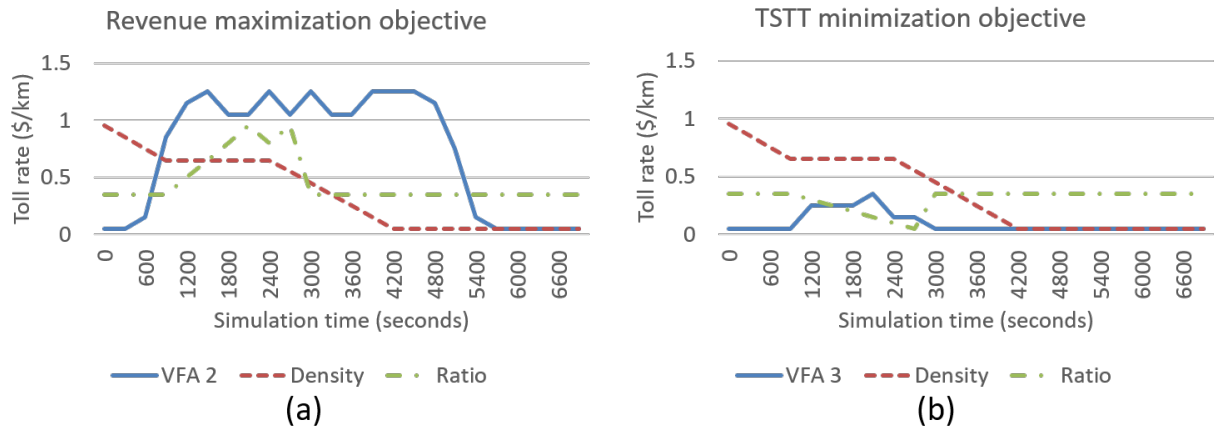


Figure 3.10: Toll profiles for the DESE network for the (a) revenue maximization and (b) TSTT minimization objectives

Figure 3.11 plots the time-space diagram showing the evolution of the ratio of current

density to jam density for each cell along the ML and GPL for both revenue-maximizing and TSTT-minimizing toll profiles. Values closer to 1 are shown in red indicating higher congestion, while the values closer to 0 are shown in green indicating lower congestion. All other values follow the spectrum in between. A close analysis of the revenue-maximizing profile shows that such policies charge a high toll value in the beginning to let the congestion build up on the GPLs. Once the GPL get congested, the policy continues to charge higher toll and attracts more traveler because of the high travel time difference created between the GPL and the ML. On the other hand, the TSTT-minimizing profile charges low toll in the beginning and thus causes both ML and GPL to become congested; however, since the managed lane needs to satisfy the speed limit constraint, the balance between the split for ML and GPL is maintained throughout the simulation.

This observation is consistent with the "jam-and-harvest" behavior of the policies maximizing the revenue, first observed in Göçmen et al. [40]. The "jam-and-harvest" behavior of the revenue-maximizing policies is a characteristic of the model and is not necessarily observed in practice. Our future work will include the search of improved policies by including a constraint that avoids congestion built-up on the GPL while no vehicle is using the ML.

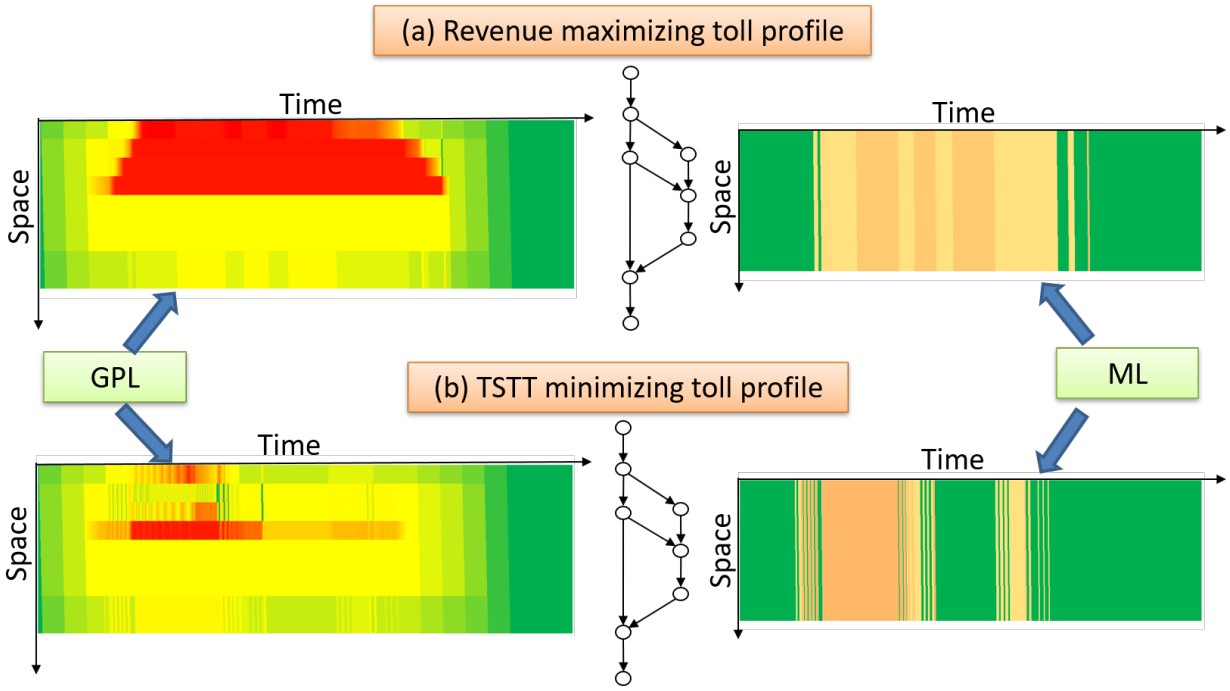


Figure 3.11: Time-space diagram showing the ratio of current density to the maximum jam-density for each cell on the GPL and the ML, for (a) revenue-maximizing toll profile, and (b) TSTT-minimizing toll profile

3.2.4 Summary

This chapter developed a formulation for determining optimal dynamic toll prices for managed lane networks with multiple entrances and exits. We proposed a definition of decision routes at each diverge point which incorporates complete set of route choices at each diverge point and scales quadratically with network size even though the total number of routes can be exponential. An optimization problem was formulated with the toll rate per km as the decision variable. Two optimization objectives were studied: revenue maximization and TSTT minimization. After making suitable assumptions, the formulation was converted to a deterministic MDP. The VFA method was used to solve the MDP problem to deal with the curse of dimensionality. The performance of the algorithm was tested on four networks using three different initializations for value functions.

The VFA method converges to within 0.2-17% of the best-found objective function for the DESE and LBJ networks. VFA2 initialization generates higher revenue than the VFA1 initialization and finds a profile with best-found maximum revenue in the first 50 iterations.

Similarly, VFA3 initialization finds a profile with the best-found minimum TSTT in the first 5–50 iterations of the VFA method. This suggests the usefulness of VFA2 and VFA3 initializations as a heuristic for dynamic pricing, suitable both for online or offline implementations. The VFA method also shows promising results in determining the toll profiles which perform better than other heuristics used in practice for both revenue maximization and TSTT minimization objectives. On an average, the heuristics predict policies generating revenues 10-90% lower than the revenues predicted by the VFA method (for revenue-maximization objective) and generating TSTT 0-27% higher than the TSTT predicted by the VFA method (for TSTT-minimization objective).

The TSTT minimization and revenue maximization objectives are found to conflict; policies generating higher revenue perform poorly on the TSTT minimization objective and vice versa. On an average, the TSTT minimization policy is found to generate revenue which is 88.6% lower than the best-found revenue and the revenue-maximizing toll profile is found to generate TSTT with is 102.3% higher than the best-found TSTT. Revenue-maximizing toll profiles are found to exhibit the "jam-and-harvest" behavior, a characteristics not observed by the profiles minimizing TSTT.

3.3 Model for distributed dynamic pricing

3.3.1 Optimization model

Assumptions

Consider a managed lane network shown in Figure 3.1(a). The upper set of links form managed lanes (ML) and the lower set of links form GP lanes. As we describe the network, we label the assumptions made in the our model as "$A\#$".

We assume that there is only one origin and destination point for all travelers in the network ($A\#1$). For the network in Figure 3.1, the origin is node O and destination node is labeled D. This assumption can be relaxed by including additional origin and destination points and by disaggregating the traffic flow based on its origin and destination; however, we adopt this assumption to simplify the explanation.

As travelers continue to travel towards the destination, they make routing decisions

at diverge locations. Nodes $1, 3, 5$, and 7 are the diverge locations for the network in Figure 3.1(a). At each diverge node, travelers use the information about the current travel time and toll values to make a lane choice decision. We assume that the information about the current travel time is provided by measuring instantaneous travel time ($A\#2$) and that all travelers have complete information about the current network state ($A\#3$).

There are two primary ways to model lane choice decision at each diverge location from the literature: using a binary logit model [10, 11] or a value of time (VOT) distribution [21, 26]. We model lane choice using VOT distribution where each traveler is assumed to have a certain value of time and they choose a path that minimizes the linear combination of toll and travel time, converted to the same units using their VOT value ($A\#4$). We further assume that at each decision point, travelers compare the current utility along a certain set of routes associated with each diverge location, which are called decision routes at each diverge location ($A\#5$). Decision routes are defined as the set of routes connecting the current diverge node to the first merge node located immediately downstream of the first exit from the managed lane if a traveler enters the lane at the current diverge node. The diverge routes for the network in Figure 3.1 are shown in Figure 3.2. We borrow this definition from [21].

Each decision point is monitored by a toll agent which controls the toll for travelers entering the managed lane at that location. We assume that the control variable is the time varying toll rate per mile where each traveler is charged a toll based on the distance traveled on the managed lane after entering the managed lane at the current decision point regardless of the exit point ($A\#6$). Other variations of the choice of control variable include charging toll rate based on entrance and exit points, or charging a constant toll rate regardless of destinations [42]. Extending our model to other variants will be a part of the future work. We also consider that travelers pay the toll rate they see while entering the managed lane and continue to pay that rate until the next decision point is reached ($A\#7$). We focus on the revenue maximization objective since it is one of the primary objectives for operating managed lanes where the funds are used to recover the costs of construction and maintain the toll facility.

The problem is formulated as a finite horizon MDP. The demand distribution at the

origin is assumed to be known ($A\#8$). The entire problem is modeled as a deterministic process ($A\#9$). This assumption is made to simplify the understanding of the results; making the problem stochastic will be a part of the future work.

Notation

We divide the time horizon into equal time steps, each one unit long. The set of all time intervals is denoted by $\mathscr{T} = \{1, 2, \ldots, T\}$, where T is the final time step. Set $\mathscr{C}$ represents the set of all cells in the network and the set of all diverging cells is denoted by $\mathscr{C}_D \subset \mathscr{C}$. We follow the Godunov scheme for discretizing the links into cells, where the length of a cell i, denoted by Δx_i, is more than the product of its free flow speed times the length of a time interval [57]. Figure 3.1(b) shows the discretized representation of the network in Figure 3.1(a).

We define a toll cell as the cell immediately after the diverge point which leads a traveler towards the managed lane. The toll is charged only for vehicles entering this cell. Set $\mathscr{C}_{\text{toll}}$ represents the set of toll cells. In the decentralized pricing model, each toll agent regulates the toll at each toll cell. A toll agent $n \in N$ manages the toll rate at the toll cell immediately following a diverge cell i.Each toll agent sets the toll using a distance based pricing model, where the toll rate per mile set by an agent n at time step t is $\beta_n(t)$. Each $\beta_n(t)$ is bounded by its minimum ($\beta_{\min}$) and maximum ($\beta_{\max}$) values.

A discrete VOT distribution is used to model lane choice. It is denoted by a set of possible VOT values $v \in V$, where the proportion of each class in the population is denoted by p_v ($\sum_v p_v = 1$). We define $x_i^v(t)$ as the number of vehicles of VOT class v in cell i at time step t and $y_{ij}^v(t)$ as the number of vehicles of class v moving from cell i to cell j from time step t to $t+1$. Throughout the rest of the chapter, variables i, n, and v are used to index variables of the set of all cells, agents, and VOT values respectively.

Multiagent Markov Decision Process Model

In this section, we explain the formulation of the toll pricing model as a cooperative MDP and its relaxed version under certain assumptions.

Complete MDP

The complete MDP problem has following parameters:

- Finite number of time steps $t \in \mathscr{T}$ and finite number of agents $n \in N$

- State vector of the system at time step t, denoted by $s(t)$. We defined $s(t)$ as a vector containing the number of vehicles of each VOT class in each cell in the network. Mathematically, $s(t) = \{x_i^v(t) \mid i \in \mathscr{C}, v \in V\}$

- Action vector at time step t, denoted by $a(t)$ and defined as $a(t) = \{\beta_n(t) \mid \forall n \in N\}$

- Deterministic transition function f determines the state at the next time step given current state and action vectors, that is $s(t+1) = f(s(t), a(t))$. The f function is governed by the traffic flow update equations in [57] where the lane choice at a diverge is determined by the value of time of a vehicle and the current travel time and toll values on each of the decision route like explained in [21]

- Reward function $R(s(t), a(t))$ determines the one step reward obtained from taking action $a(t)$ in state $s(t)$. For the revenue maximization problem, the reward is the product of the number of vehicles choosing the managed lane times the toll rate per mile times the length of travel on the managed lane. Additionally, since the managed lane is to kept uncongested at all times, we penalize tolls which push more vehicles towards managed lanes than required with a reward of -100

The objective of the model is to find a policy $\pi : s(t) \rightarrow a(t)$ which maximizes the total sum of one step reward across all time steps and agents, given the initial state $s(0)$. Since the one step reward depends on the joint action of all agents, each agent has to collaborate with the others to obtain an optimal policy. We define the optimal value of being in a state $s(t)$ by value functions $V^*(s(t))$ which represent the total reward obtained from starting in state $s(t)$ at time t and choosing optimal actions thereafter. At optimality, the value functions satisfy the Bellman equation (3.13):

$$V^*(s(t)) = \max_{a(t)} \{R(s(t), a(t)) + V^*(s(t+1))\} \tag{3.13}$$

Relaxed MDP

Solving optimal policies in a multiagent setting where the actions are a continuous function of time is a challenging task. The regular Q-learning or value function approximation

methods fail due to the curse of dimensionality, where the computation time is exponential in the number of agents. In the case where toll agents collaborate and need to coordinate their actions with few "neighboring" agents only, we can approximate the value function of a state as the sum of value functions defined for each agent as shown in (3.14):

$$V^*(s(t)) = \sum_{n \in N} V_n^*(s_n(t)) \tag{3.14}$$

,where, $V_n^*(s_n(t))$ is the value function associated with agent n defined at a local state vector $s_n(t)$. This value function denotes the total future reward obtained by agent n starting from local state $s_n(t)$ at time step t and assuming all agents take joint optimal actions thereafter. The local state vector for agent n is defined as the number of vehicles of each VOT class in each cell located along the decision route for the diverge cell associated with agent n. Substituting (3.14) in (3.13) for both $s(t)$ and $s(t+1)$, and decomposing the reward function as the sum of reward function for each agent ($R_n(s(t), a(t))$), we can write a new form for the Bellman equation decomposed for each agent, similar to the Q function decomposition in [62]:

$$V_n^*(s_n(t)) = \max_{a(t)} \{R_n(s(t), a(t)) + V_n^*(s_n(t+1))\} \ \ \forall n \in N \tag{3.15}$$

3.3.2 Solution methods

To solve the relaxed MDP model, we use a variant of the sparse cooperative Q-learning algorithm from Kok and Vlassis [55], where we replace learning Q-functions for each agent and state with learning value functions for each local state for each agent. We call this algorithm `SparseV`. The algorithm estimates the value function, $V_n(s_n(t))$ for each agent and at each time step. The basic structure of the algorithm is presented in Algorithm 1. The algorithm begins with an initialization of the value functions and then simulates a policy generated using the current estimates of value functions. It uses the ϵ-greedy approach for policy selection with a decreasing value of ϵ to balance exploration and exploitation. Next, it changes the estimates of the value functions of the visited states by combining it with the

current estimates using a step size which decreases harmonically with time. The process is repeated until the convergence of value functions or till a maximum number of iterations. A superscript m on $V_n^m(s_n(t))$ indicates the iteration number.

Algorithm 4 `SparseV` using look-up table representation

Step 0**:** Initialization
Initialize $V_n^0(s_n(t)) \quad \forall n \in N, t \in \mathscr{T}, s_n(t)$
Choose initial state $s(0)$ and set $m = 0$

Step 1**:** Simulating a policy
Set $\widehat{V}_n^m(s_n(t))$ =null $\forall s_n(t)$
for $t \in \{1, 2, \ldots, n(\mathscr{T})\}$ **do**
  if random number between 0 and 1 less than ϵ **then**
    Select $a(t)$ randomly between $\max\{\beta_{\min}, a(t-1) - \$0.25\}$ and $\min\{\beta_{\max}, a(t-1) + \$0.25\}$
  else
    $a(t) =$ `localPolicySearch`$(a(t-1),$
    $V_n^m(s_n(t)))$
  end if
  Determine $s(t+1) = f(s(t), a(t))$ and $s_n(t+1)$
  Determine the updated value function estimate for state $s_n(t)$:
  $\widehat{V}_n^m(s_n(t)) = R_n(s(t), a(t)) + V_n^m(s_n(t+1)))$
end for

Step 2**:** Update the V values for the visited states
Step size update: $\alpha_m = 20000/(20000 + m)$
for $t \in \{T, \ldots, 3, 2, 1\}$ in the reverse order of time and **for each** agent $n \in N$ **do**
  if $\widehat{V}_n^m(s_n(t))$ is NOT null **then**
    $V_n^{m+1}(s_n(t)) = (1 - \alpha_m)V_n^m(s_n(t)) + \alpha_m \widehat{V}_n^m(s_n(t))$
  end if
end for
if $m >$ max number of iterations or if V values converged **then**
  Stop. Report $V_n^m(s_n(t))$ as the final value estimates
else
  $m \leftarrow m + 1$ and go back to Step 1
end if

To find an optimal joint action given the current value function estimates, we assume that the action of an agent is influenced only by its "downstream neighboring agents" ($A\#10$). We define downstream neighboring agents as agents located downstream of the current agent which lie on the decision routes associated with the current agent's diverge cell. This assumption is reasonable since the toll values set by all downstream neighboring agents

immediately impacts the decisions made by the travelers at current agent's toll gantry.

We show this influence relationship using a directed coordination graph (CG) corresponding to the managed lane network. The nodes of a CG represent the agents and edges connect agents which are assumed to influence actions of each other. If an edge is directed from agent n_1 towards agent n_2, then the action of agent n_1 influences the action of agent n_2. Given the managed lane network is acyclic, the CG is also acyclic and thus has a topological order. For the managed lane network in Figure 3.1, the CG is shown in Figure 3.12. In contrast to the undirected coordination graph approach used in the literature [49, 52, 55], we choose a directed CG to simplify the local policy search.

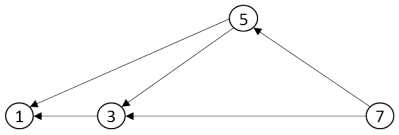


Figure 3.12: Coordination graph for the network in Figure 3.1 with agents as nodes and edges connecting agents representing interdependencies

The `localPolicySearch()` method solves the optimal action of each agent given the current value function estimates (shown in detail in Algorithm 2). It visits agents in the topological order of the CG and determines the optimal action assuming the action of all downstream neighboring agents is fixed. It first finds the threshold toll values for each agent corresponding to each VOT class (β_n^v). The calculation of these threshold values is explained later. Next, it evaluates the gain g_n^v for agent n for each threshold toll β_n^v and sets the action of the agent to the threshold toll that results in the maximum gain. We define gain as the sum of the total one step revenue for that agent and the value of the resulting next state from the joint action. Unlike the approach in Rezaee [49] and El-Tantawy et al. [52] where the action of agents are continuously changed till no agent can cause any gain by changing its action (the stopping point for which is not guaranteed), our approach terminates after one sweep of the CG and thus the computation time is linear in the number of agents.

The toll threshold values for each agent enable the search on a continuous action space. We determine these thresholds by exploiting the fact that there are only a finite number of VOT classes and thus only finite toll values can lead any VOT class to choose the managed lane. We demonstrate the evaluation of threshold tolls using an example.

Algorithm 5 `localPolicySearch`$(a(t-1), V_n^m(s_n(t)))$

Set $a(t) = a(t-1)$
for agent n in the topological order of the CG **do**
  Determine β_n^v for all $v \in V$
  for $v \in V$ **do**
    Set $\beta_n(t) = \beta_n^v$ and determine gain:
    $g_n^v = R_n(s(t), a(t)) + V_n^m(s_n(t+1))$
  end for
  Let $\bar{v} = \text{argmax}_v\ g_n^v$. Set $\beta_n(t) = \beta_n^{\bar{v}}$
end for
Return $a(t)$

Consider the diverge node 5 on the network in Figure 3.1. There are three paths over which a traveler compares the utility using instantaneous travel time and toll values. Table 3.2 shows these paths, and the instantaneous travel time, the toll, and the total disutility for a vehicle with VOT value v for each path. The route $\{5, 8, 9, 10\}$ leads a traveler towards the managed lane.

Table 3.2: Disutility comparison over decision routes for agent 5 *for a vehicle of value of time class* v

Decision route	Inst. travel time	Inst. toll	Total disutility
$\{5, 8, 9, 10\}$	τ_1	β_1	$\beta_1 + v\tau_1$
$\{5, 6, 7, 8, 9, 10\}$	τ_2	β_2	$\beta_2 + v\tau_2$
$\{5, 8, 7, 10\}$	τ_3	0	$v\tau_3$

The objective of the action selection method is to determine the toll rate β_1, given the instantaneous value of travel times τ_1, τ_2, and τ_3, and the assumed fixed value of β_2. If the VOT values are defined using a discrete distribution ($v \in V$), then the value of β_1 which lets vehicles of VOT class v onto the managed lane is the one which causes the route $\{5, 8, 9, 10\}$ to have the minimum disutility. That is, if we define the threshold toll value corresponding to VOT class v (β_1^v) as in (3.16) and (3.17), then for all $\beta_1 < \beta_1^v$, all vehicles of VOT class v will choose the managed lane. We also ensure that the threshold value belongs to the feasible toll values by bounding it between the limits $\beta_{\min}$ and $\beta_{\max}$. This method reduces

the search on a continuous action space to a search over finite β_1^v values.

$$\beta_1^v = \min\{\beta_2 + v(\tau_2 - \tau_1), v(\tau_2 - \tau_1)\} \tag{3.16}$$

$$\beta_1^v = \min\{\beta_{\max}, \max\{\beta_{\min}, \beta_1^v\}\} \tag{3.17}$$

We compare the performance of the `SparseV` algorithm against following heuristics. The first two are feedback control based heuristics which seek to maintain the traffic flow operation on the managed lane at the desired level and are commonly used in the field implementations. The last heuristic generates the toll profiles randomly.

1. Density based heuristic (`Density`): Using this heuristic, each toll agent monitors the density on the managed lane cells (defined as the ratio of the current number of vehicles in the cell to the maximum number of vehicles allowed in the cell) downstream of the current diverge cell operated by the agent. If the density is different from the desired density, the toll is increased or decreased using a regulator parameter.

2. Ratio based heuristic (`Ratio`): Similar to the `Density` heuristic, each toll agent monitors the ratio of the density on the managed lane cells to the density on the GP cells downstream of the current diverge cell operated by the agent. If the ratio is different from the desired ratio, the toll is increased or decreased using a regulator parameter.

3. Random search (`Random`): We simulate 100,000 random policies where the action of each agent is chosen randomly and select the policy which generates highest revenue.

3.3.3 Experiments

We test the performance of the algorithms on two networks shown in Figure 3.13. The first network has double entrances and a single exit (DESE) with two agents located at each entrance point, while the second network have three entrances and two exits with four agents located at the three entrances and the first exit. The second network is an approximation of the 3.5-mile long toll segment 2 of the LBJ TEXpress lanes in Dallas, TX.

We consider five VOT classes with VOT values as \$10/hr, \$15/hr, \$20/hr, \$25/hr, and \$30/hr, with assumed known proportions of demand as $0.1, 0.4, 0.2, 0.2$, and 0.1 respectively.

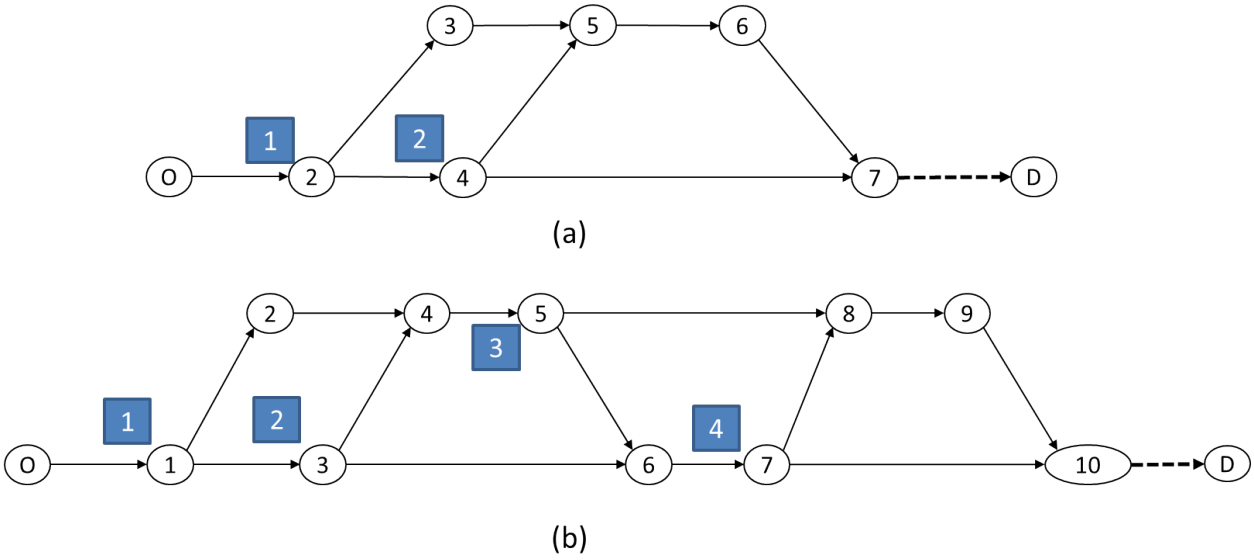


Figure 3.13: Two test networks: (a) DESE network; (b) LBJ TEXpress toll segment 2 abstract network. The dashed link indicates a bottleneck

The traffic flow follows a trapezoidal fundamental diagram with free flow speed as 60 mph, back wave speed as 20 mph, link capacity as 2200 veh/hr/lane, and jam density as 265 veh/mile. Each time step is assumed 6 seconds long. The minimum and maximum values of toll rate are set as \$0.1/mile and \$3/mile respectively. The network is simulated for 30 minutes with no initial congestion. Buildup of congestion is modeled using a downstream bottleneck located at the end of each network. The value functions in the `SparseV` algorithm are initialized by an upper bound revenue, as shown in (3.18), where q_n and l_n are respectively the capacity and the length of the toll cell following the diverge cell associated with agent n.

$$V_n^0(s_n(t)) = (q_n \beta_{\max} l_n)(T - t) \tag{3.18}$$

DESE network

To test that the algorithm converges when the time horizon is short, we simulated the DESE network with initial congestion on the GP lanes for 1 minute. Figure 3.14 shows the variation of the moving average revenue (calculated as the average revenue over last 10 iterations) with the iteration number. As observed, the average revenue converges to a stable state after approximately 1200 iterations, after which the moving average is only influenced by the spikes generated by the random policies simulated using the ϵ-greedy approach. The

best revenue maximizing policy for this network generated a revenue of \$63.5 in 1 minute, while the revenues generated by the `Density`, `Ratio`, and `Random` heuristics were \$16.8, \$12.5, and \$53.35 respectively. At convergence, the `SparseV` method generated a revenue of \$56.2, which is still better than the other heuristics, though not optimal.

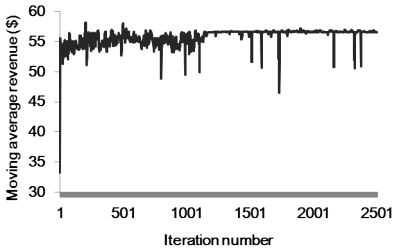


Figure 3.14: Convergence of `SparseV` method on DESE network for 10 timesteps

Figure 3.15(a)-(d) show the results for the DESE network for 30 minutes of simulation. Figure 3.15(a) shows the moving average revenue with iterations. We observe that the convergence of the `SparseV` method is not guaranteed for longer time horizons. This can be explained by the graph in Figure 3.15(b) which shows the number of new states visited in each iteration. The graph starts flattening out towards the later half of the simulation at a value around 100, that is, the `SparseV` method is still exploring an average of 100 new states in the later iterations across both agents. Convergence can be expected when new states are not explored and instead, the values of the older states are updated. Nevertheless, in the process of iterating the `SparseV` method, the toll profiles which led to the highest revenue are shown in Figure 3.15(c) and (d) for agents 1 and 2, respectively.

Table 3.3 compares the best revenue obtained from the simulated policies. As observed, the revenues generated by the `Density` and `Ratio` heuristics are $70 - 75\%$ percent lower than than of the `SparseV` algorithm. The `SparseV` generates revenue which is 9.42% lower than the best revenue obtained by the `Random` algorithm; however it only takes 3 minutes of simulation time on a 2.8Ghz 64-bit Windows machine, in contrast to the 8 minutes of computation for the `Random` policy. We also observe that the `Density` and `Ratio` based heuristics lead to an average of 37% violations (defined as the proportion of the simulation time period the managed lane is congested) on the managed lane where additional 605 and 661 vehicles are let onto the managed lane causing the speed in the lanes to fall below the

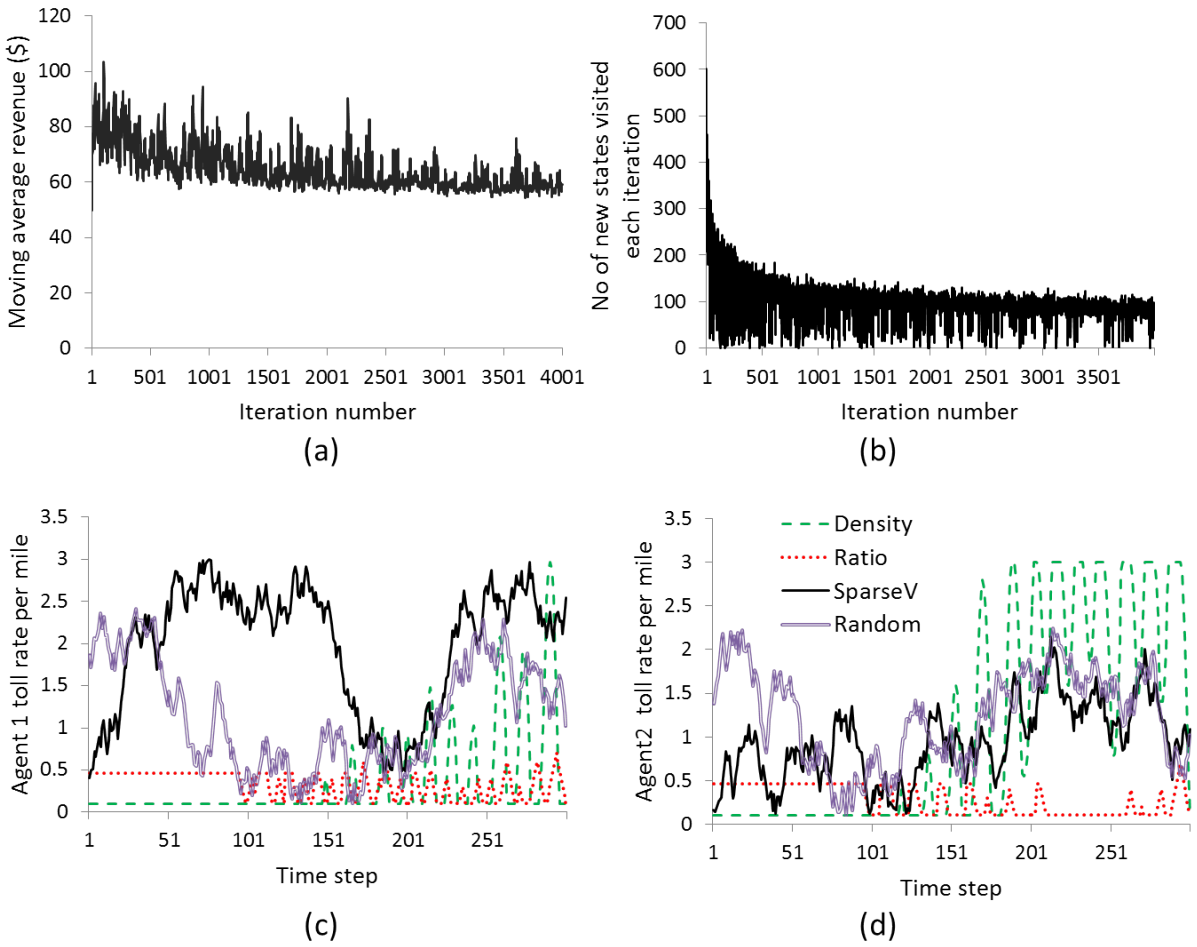


Figure 3.15: Tests on DESE network. (a) Converge rate of the ***SparseV*** *method with iterations; (b) Number of new states explored each iterations; Agent 1 (c) and Agent 2 (c) toll rate with time*

free flow speed. The best toll policy from `SparseV` only causes 5% violations.

Better performance of the toll policies generated by `Random` and `SparseV` algorithms can be explained by the jam-and-harvest nature of the optimal policies [21, 40]. As shown in Figure 3.15(c) and (d), the `SparseV` and `Random` policies charge higher toll in the earlier time steps to let the GP lanes become congested ("jam") and then continue charging higher toll rate in the later time steps to obtain more revenue when there is higher demand trying to enter the facility ("harvest"). This behavior of the optimal revenue policies is a characteristic of our model and can be avoided in practice using regulations on toll changes, studying which will be a part of our future work.

Overall, the results show that the `SparseV` method is successful in predicting better

Table 3.3: Comparison of revenues across different algorithms for DESE network

Algo.	Max. revenue	% Violations	Extra vehicles on ML
`Density`	38.32	37.7%	605.00
`Ratio`	33.09	36.6%	661.00
`Random`	146.00	0	0
`SparseV`	132.25	5%	38

policies than the other heuristics used in practice, though the policies may not converge to optimal and may exhibit undesirable characteristics like "jam-and-harvest".

LBJ network

Figure 3.16 and Table 3.4 show the performance of the four algorithms on the LBJ test network. As observed, the `SparseV` algorithm's best toll policy generates 24.3% more revenue than the best policy generated by the `Random` method, and $75-86\%$ more revenue than the `Density` and `Ratio` heuristics. The `Density` and `Ratio` heuristics continue to perform worse due to their inability to coordinate the tolls between agents.

Table 3.4: Comparison of revenues across different algorithms for LBJ network

Algo.	Max. revenue	% Violations	Extra vehicles on ML
`Density`	125.21	32.33%	1313.00
`Ratio`	109.63	32.0%	3358.00
`Random`	602.38	0%	0
`SparseV`	795.32	0%	0

The better performance of the `SparseV` algorithm can be explained by the coordination of tolls between the agents. As observed in Figure 3.16, the `SparseV` algorithm strategically charges lower toll for agent 2 at earlier time steps such that more vehicles are diverted towards the managed lane at that entrance, and once these vehicles arrive the diverge point for agent 3, they are faced with a higher travel time savings on the managed lane because the GP lane is congested due to the spillback from the downstream bottleneck. Thus, agent 3 can charge higher toll in later time steps to generate higher revenue. The same is true for agent 4 charging higher toll towards the later half of the simulation.

Figure 3.17 shows the convergence of the `SparseV` algorithm with iterations. Con-

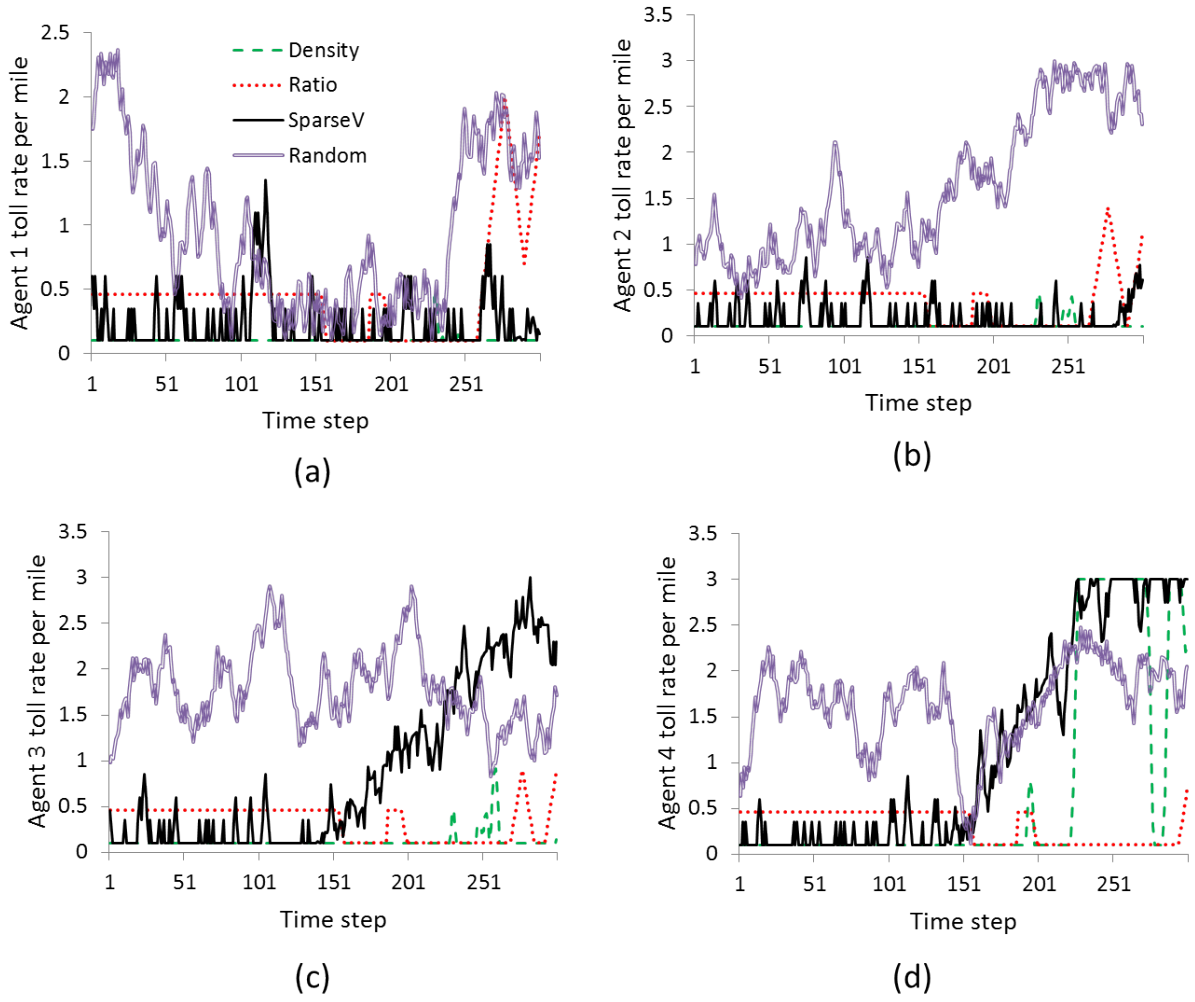


Figure 3.16: Toll profiles for the 4 agents compared for each of the four algorithms

sistent with the observations on DESE network, `SparseV` fails to converge within 1500 iterations and continues to oscillate around a certain toll revenue. Overall, we observe that the `SparseV` method does well in predicting policies which do better than the `Density` and `Ratio` heuristics, but it shows a lack of convergence.

3.3.4 Summary

In this section, we proposed a multiagent reinforcement learning algorithm for the dynamic pricing of managed lanes with multiple entrances and exits where each agent regulates its own toll and coordinates with other agents to optimize the system performance. We focused on revenue maximization as our objective. Our future work will focus on extending

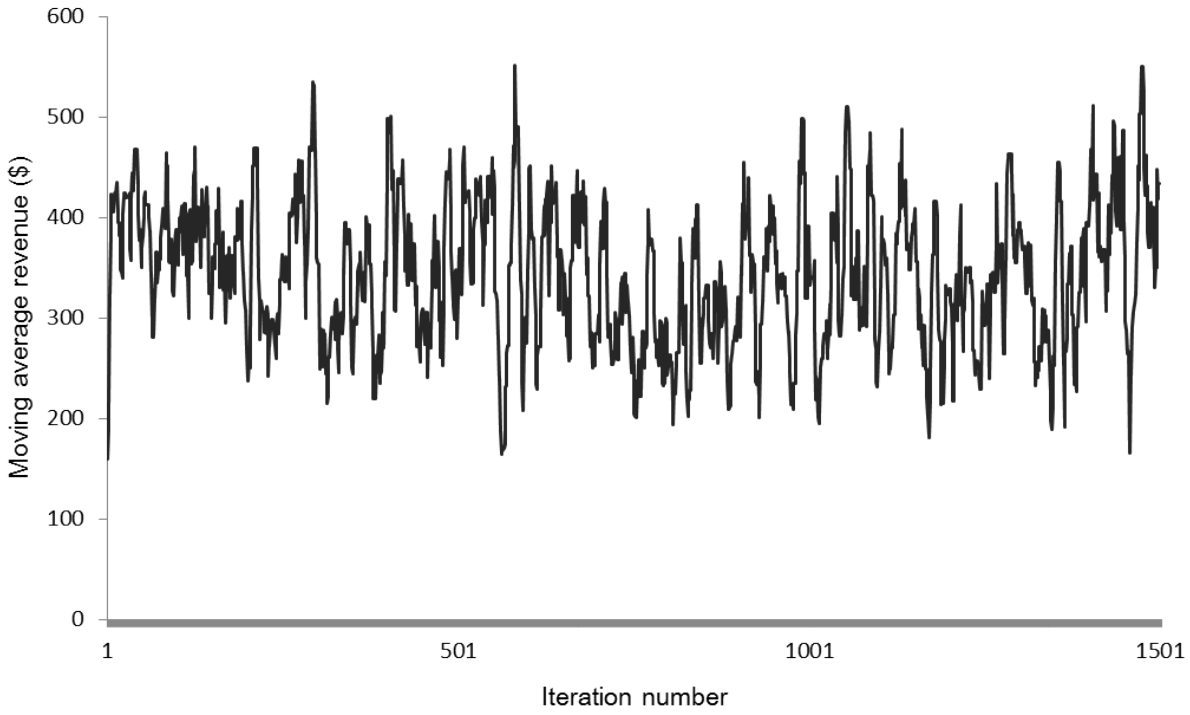


Figure 3.17: Convergence of the ***SparseV*** *algorithm on the LBJ network*

this work towards other objectives like maximizing throughput, minimizing delay or total system travel time, or a combination of these. The proposed `SparseV` algorithm builds on the assumption that the total value function in any state can be decomposed into value function for each agent. We proposed a `localPolicySearch` method to determine the tolls using a directed coordination graph modeling interactions between the agents. The method chooses joint optimal action by exploring the continuous action space.

Our experiments on two test networks show promising results. The `SparseV` algorithm performed better than the `Density` and `Ratio` heuristics by generating revenues $70\% - 86\%$ higher than those heuristics. `SparseV` also did comparably well to the `Random` heuristic, producing revenues within $9 - 20\%$ of the heuristic. `SparseV` has an advantage over the `Random` heuristic that it takes less computation time and does not enumerate toll policies, but rather builds on a MDP structure.

Though the `SparseV` method shows promising results, it has several limitations which need to be addressed. The first is the issue of convergence where the algorithm continues to oscillate heavily. This issue is inbuilt in all Q-learning based algorithms because they

depend heavily on the Q or value functions initialization. Second, the algorithm is shown to converge to values which are suboptimal. This lack of convergence to the optimal depends on the aggregation level used for the state space. Third, the jam-and-harvest nature of the optimal policies is not desired in practice and thus constraints on toll policies to prohibit this nature needs to be modeled. In the next chapter, we propose a deep reinforcement learning framework to overcome these limitations.

Chapter 4

Deep Reinforcement Learning Algorithm for Dynamic Pricing

In this chapter, we take an artificial intelligence approach to dynamic pricing of express lanes. Dynamic pricing for MLs with multiple access locations is a complex control problem due to the heterogeneity in lane choice behavior of travelers with varying values of time and destinations of travel. Predicting driver behavior with certainty is difficult. A recent study showed that a binary logit model, commonly used for modeling lane choice, is inadequate in predicting heterogeneity in lane choice observations [25].

Several dynamic pricing algorithms have been explored in the literature that optimize tolls under varying assumptions on driver behavior. These include methods using stochastic dynamic programming [11], hybrid model predictive control (MPC) [12, 38], reinforcement learning (RL) [10, 63], and approximate dynamic programming [21]. While these algorithms do well against existing heuristics, they make some or all of the following restricting assumptions, which we relax:

1. Restricted access for travelers: travelers do not exit the managed lane once they enter till their exit is reached [10, 11], and that only the first entry location is considered for lane-choice decision [12]

2. Fully observable system: toll operators have access to measurements of traffic density throughout the network for optimizing tolls [10, 11, 12, 21, 63]

3. Ignored traveler heterogeneity: a single vehicle class is considered with a single origin and destination [10, 11, 21]

4. Simplified traffic dynamics: for example, the flow dynamics on general-purpose lanes (GPLs) are assumed independent of vehicles using the ML [11]; or the proportion of flow split at diverge points is assumed identical for all origins [12]

In addition, there are relatively few analyses on the conflict between optimization of multiple objectives with realistic constraints. In Chapter 3 showed that the revenue-maximizing tolls exhibit a *jam-and-harvest* (JAH) nature where GPLs are intentionally jammed to congestion earlier in the simulation to harvest more revenue towards the end. Handling such undesirable behavior of optimal policies has not been studied in the literature.

Furthermore, practical applicability of these algorithms in real-world environments is a less-explored question. Algorithms that optimize prices using a simulation model can be applied in real time using lookup tables. However, the transferability analysis of such lookup tables to new input distributions is not considered [10, 11, 21]. The hybrid MPC algorithm in Tan and Gao [12] follows a different procedure for practical applications. It predicts boundary traffic as an exogenous input using a simulation model and optimizes tolls over a finite horizon using real-time measurements of traffic densities and queue lengths. However, solving an MPC-based model with heterogeneous vehicle classes and partial observability of the system, without the restricting assumptions stated earlier, is complex and not fully studied. We thus require scalable algorithms for real-world networks that relax the assumptions on driver behavior and traffic flow, and transfer well from simulation settings to new input distributions.

In this chapter, we focus on pricing algorithms that rely on real-time density observations using sensors (such as loop detectors) located only at certain locations around the network without access to any information about the demand distribution or driver characteristics like the value of time (VOT) distribution. We use deep reinforcement learning (Deep-RL) algorithms for optimizing tolls while relaxing simplifying assumptions in the earlier literature. In the recent years, Deep-RL algorithms have been successfully used for applications such as playing Atari games and planning the motion of humanoid robots like `MuJoCo` [64]. Similar algorithms have been applied for traffic signal control [65], active traffic management [50], and control of autonomous vehicles in mixed autonomy [66].

Simply applying Deep-RL as a "black box" is unlikely to yield effective solutions for pricing dynamic lanes, due to the size of the state space and the potential for undesirable jam-and-harvest behavior. We have introduced two domain-specific elements into our formulation and experiments: the use of a "decision route model" as a concise (polynomial-space) way

of simulating route choice in corridors with multiple ML entrances and exits; and the use of reward shaping to avoid JAH phenomena. In addition, using Deep-RL algorithms, we relax assumptions in the literature by considering multiple origins and destinations, multiple access points to the managed lane facility, *en route* diversion of vehicles at each diverge point, and partial observability of traffic state.

The key contributions of this chapter are:

- We demonstrate the usefulness of Deep-RL algorithms for solving dynamic pricing control problem under partial observability, and show that it performs well against existing heuristics, without requiring restricting assumptions on driver behavior or traffic dynamics.
- We apply multi-objective optimization methods for joint optimization of multiple objectives and overcome undesirable JAH characteristics of revenue-maximizing optimal policies.
- We conduct tests to verify the transferability of learned Deep-RL algorithms to new input distributions and make recommendations on real-time implementation of the algorithm.
- We develop an open-source framework for dynamic pricing using multiclass cell transmission model available for benchmarking future dynamic pricing experiments.

4.1 Literature review

Many control problems have been studied in the area of transportation engineering including active traffic management strategies such as ramp metering, variable speed limits, dynamic lane use control, and adaptive traffic signal control (ATSC). These control problems can be broadly solved using three methods: open-loop optimal control methods (that solve the optimal control problem without incorporating real-time measurements), closed-loop control methods like MPC (that incorporate the feedback of real-time measurements and optimize over a rolling horizon), and lately RL methods where the optimal control is learned with an iterative interaction with the environment, possibly in simulated offline settings which can then be translated in real world settings. Refer Ferrara et al. [67, Chapter 8] for an overview of control problems in the transportation domain.

The managed lane pricing problem is also a traffic control problem, where the chosen control directly impacts the driver behavior and thus the congestion pattern. There are three component models to the ML pricing problem [26]: a *lane choice model* that determines how travelers choose a lane given the tolls and travel times, a *traffic flow model* that models the interaction of vehicles in simulated environments, and a *toll pricing model* which determines the toll pricing objectives and how the optimization problem is solved to achieve the best value of the objective. Pandey [68] presented a tabular comparison of component models for the existing models in the literature. In this research, we focus on the *toll pricing models*.

Toll pricing models for MLs with a single access point are commonly studied. Gardner et al. [26] argued that for MLs with a single entrance and exit, the tolls minimizing the total system travel time (TSTT) also utilize the managed lanes to full capacity at all times. The authors developed an analytical formulation for tolls minimizing TSTT which send as many vehicles to the ML at each time step as is the capacity of the lane. Lou et al. [27] used a self-learning approach for optimizing toll prices where the average VOT values were learnt using real-time measurements. Toledo et al. [38] used a rolling horizon approach to optimize future tolls with predicted demand from traffic simulation; however, the method of exhaustive search to solve the non-convex control problem does not scale well for large managed lane networks.

For managed lanes with multiple access points, Tan and Gao [12] presented a formulation where the proportion of vehicles entering the managed lane is optimized instead of directly optimizing the toll prices. The authors showed a one-to-one mapping between optimal toll prices and the proportion values, and transformed the control problem into a mixed-integer linear program which can be solved efficiently for networks with multiple access points. Dorogush and Kurzhanskiy [43] used a similar method and optimized split ratios at each diverge, which are then used to determine toll prices; however, their analysis ignored the variation of incoming flow at each diverge. Apart from these optimal control based methods, Zhu and Ukkusuri [10] and Pandey and Boyles [21] used RL methods, where the control problem is formulated as a Markov decision process (MDP) and the value function (or its equivalent Q-function) is learned by iterative interactions with the environment. However, the tests are conducted for discrete state and action spaces assuming full observability of

the system. The present chapter is guided by advances in RL methods, and improves these earlier RL-based approaches for dynamic pricing.

Deep-RL improves traditional RL by using deep neural networks as function approximators, which has been effective in various control problems. See Arulkumaran et al. [64] for a survey of Deep-RL applications. Application of Deep-RL algorithms for traffic control problems is not new. Belletti et al. [50] developed an "expert-level" control of coordinated ramp metering using Deep-RL methods with multiple agents and achieved precise adaptive metering without requiring model calibration that does better than the traditional benchmark algorithm named ALINEA. Wu et al. [66] used Deep-RL algorithms to solve the control problem of selecting the acceleration and brake of multiple autonomous vehicles (AVs) under conditions of mixed human vehicles and AVs to mitigate traffic congestion. When compared against classical approaches, their approach generated 10-20% lower TSTT. Other applications of Deep-RL algorithms are in the domain of ATSC including traditional one signal control [65, 69], coordinated control of traffic signals [70], and large-scale multiagent control using Deep-RL methods [71]. See Yan et al. [72] for a review of RL algorithms in the area of ATSC.

4.2 Model for deep reinforcement learning

4.2.1 Network notation

Consider the directed network shown in Figure 4.1 which is an abstraction of a managed lane network. The upper set of links form MLs, the lower set of links form GPLs, and the ramps connect the two lanes at various access points. As we describe the network, we label the assumptions made in our model as "A#". We also label ideas for future work as "FW#".

Let N represent the set of all nodes and $A = \{(i, j) \mid i, j \in N\}$ represent the set of all links in the network. Let N_o denote the set of all origins and N_d denote the set of all destinations. We assume that origins and destinations connect to the network through nodes on the GPLs (A#1) and the only way to access the MLs is through on-ramps leading towards the lane. This is a reasonable assumption as most current ML installations allow

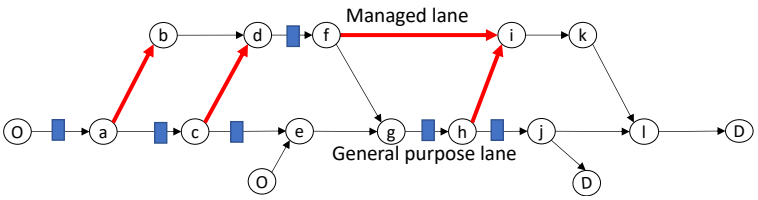


Figure 4.1: Managed lane network with multiple entrances and exits where links with higher thickness are tolled, and links with a box are observed by the toll operator

access to MLs only through ramps from the GPL. If there is a direct access to the ML from outside the network, the current framework can still be used by appropriately adjusting the lane choice model explained in Section 4.2.2.

The time horizon is divided into equal time steps, each Δt units long. The set of all time periods is given by $\mathscr{T} = \{t_0, t_1, t_2, \ldots, t_{T/\Delta t}\}$, where T, an integral multiple of Δt, is the time horizon. Tolls are updated after every $\Delta\tau = m\Delta t$ time units, where m is a positive integer fixed by the tolling agency. Define $\mathscr{T}_\tau = \{k \mid t_{km} \in \mathscr{T}$, where $k \in \{0, 1, 2, \ldots\}\}$ as the set of time periods where tolls are updated, indexed in increasing order of positive integers. Then, $|\mathscr{T}_\tau| = T/\Delta\tau + 1$. For example, Figure 4.2 shows different elements of time where $m = 4$ and $T = 16\Delta t$. For the figure, $\mathscr{T} = \{t_0, t_1, t_2, \ldots, t_{16}\}$ and $\mathscr{T}_\tau = \{0, 1, 2, 3, 4\}$.

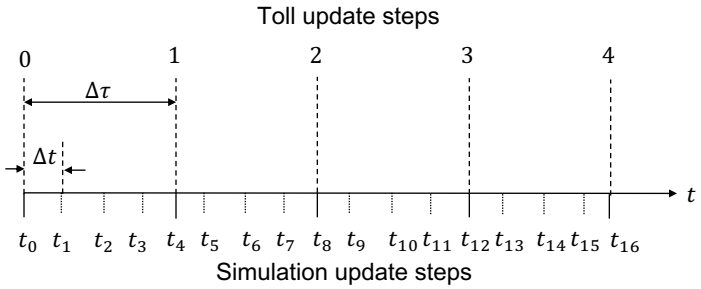


Figure 4.2: Representation of a time scale

The demand between an origin and a destination is a random variable. A toll operator does not know the demand distribution, but only relies on the observed realizations of demand. However, for simulation purposes, we model the demand of vehicles from origin $r \in N_o$ to destination $s \in N_d$ at time $t \in \mathscr{T}$ to be a rectified Gaussian random variable with mean $d_{rs}(t)$ and standard deviation σ_d, and ignore correlations of demand between different origin-destination (OD) pairs and across time. The mean demand $d_{rs}(t)$ can be estimated

by observing the historical data of the managed lane facility or from the regional model.

Let V denote the set of all values of VOT (assumed to be a discrete distribution for the population, A#2) and p_v be the proportion of demand with VOT v, for any $v \in V$. The p_v values are unknown to a toll operator. For simulation purposes, we choose the VOT distribution $(p_v \mid v \in V)$ and σ_d to be identical for all origin-destination pairs. Though dynamic traffic assignment models have been used in the literature for optimization of toll prices for MLs [73], we focus on real-time optimization of toll prices and ignore route-choice equilibration of travelers (A#3). The lane choice models are discussed in Section 4.2.2.

Traffic flow models can either be microscopic or macroscopic. With the exception of Belletti et al. [50], all other Deep-RL models in transportation domain use microsimulation to capture the vehicle-to-vehicle interactions. In this chapter, we use macroscopic models to represent traffic flow for the simplicity they provide. In contrast to the cell-based representation of managed lane network in macroscopic traffic models from the literature, where MLs and GPLs are modeled as part of the same cell [11, 12, 43], we divide each link into individual cells, where the links for GPLs are separate from that of MLs. This choice lets us use the cell transmission model (CTM) equations from Daganzo [57] for modeling traffic flow. Let $\mathscr{C}_{(i,j)}$ represent the set of all cells for link $(i,j) \in A$ and $\mathscr{C} = \bigcup_{(i,j)\in A} \mathscr{C}_{(i,j)}$ denote the set of all cells in the network. The length of each cell $c \in \mathscr{C}$, denoted by l_c, is determined as usual (the distance traveled at free flow in time Δt) [57], and is assumed constant for all links in the network (A#4). We thus require all link lengths to be integral multiples of the cell length. Let $l_{ij}, \nu_{ij}, q_{\max,ij}, w_{ij}$, and $k_{\text{jam},ij}$ represent the length, free-flow speed, capacity, back-wave speed, and jam density, respectively, for link $(i,j) \in A$ as its fundamental diagram parameters, which we assume has a trapezoidal shape (A#5).

A toll operator is assumed to manage the toll rate at each on-ramp and diverge point beyond a diverge on a ML (A#6). We assume this toll structure in contrast to the generic structure of separate toll values for each origin-destination (OD) pair, like in Yang et al. [11] and Tan and Gao [12], because it inherently models the constraint that traveling longer distance on the ML levies a higher toll than traveling shorter distance. For a detailed discussion on various options to charge toll on a managed lane network with multiple accesses, see Chapter 2 [74]. Let A_{toll} represent the links where tolls are collected. Figure 4.1 highlights

these links in bold. We denote the toll charged on link $(i, j) \in A_{\text{toll}}$ for any $t \in \mathscr{T}$ by $\beta_{ij}(t)$.

4.2.2 Lane choice model

Travelers make lane choice decisions at each diverge location (nodes a, c, f, and h in Figure 4.1), where information about the current travel time and toll values is displayed. We assume that the information about the current travel time is provided by measuring instantaneous travel time (A#7), and that all travelers make their lane choice decision only using the instantaneous/real-time information (A#8). Assumptions A#7 and A#8 are only made for simulation purposes, as the Deep-RL model only requires the realization of lane choices in form of observed loop detector measurements. If we have an estimate of experienced travel time on each route, the simulations can be based on experienced travel time. Assumptions A#3 and A#8 are related: because we assume no prior experience for the drivers, users do not find an equilibrium over route choices. Considering dynamic route-choice equilibrium while optimizing a dynamic stochastic control is a complex problem and will be studied as part of the future work (FW#1).

Conceptually, the lane choice models can be categorized based on three characteristics: the number of routes over which travelers compare the utility, whether or not the lane choice is stochastic/deterministic, and the heterogeneity in vehicles' value of time (single class vs multiple classes). Commonly used binary Logit model assumes stochastic lane choice over two routes connecting current diverge to the destination, while the decision route model evaluates deterministic lane choice of multiple vehicle classes comparing utilities over a set of routes connecting current diverge to the merge after the first exit from the ML [21]. The analysis in Chapter 2 showed that the decision route model has least error compared to the optimal route choice model for rational travelers and offers an efficient way for simulating route choices over a corridor, thus providing a potential for speeding up Deep-RL training.

Table 4.1 shows the combinations of categories and models used in the literature. Certain combination have not been used directly, but they could be used. For example, combining decision routes with stochastic lane choice can result in models like multinomial logit or mixed logit, but the assumption that the utilities across overlapping routes are independent may not hold true.

Table 4.1: Categorization of lane choice models for managed lanes with multiple entrances and exits

Number of VOT classes	**Number of routes over which the utility(s) is (are) compared**	**Deterministic or Stochastic**	**Reference(s) in the literature using this lane choice**
Single	Two	Deterministic	[26]
Single	Two	Stochastic	[12][38][10][11]
Single	Decision routes	Deterministic	None
Single	Decision routes	Stochastic	None
Multiple	Two	Deterministic	[41]
Multiple	Two	Stochastic	None
Multiple	Decision routes	Deterministic	[21],[63]
Multiple	Decision routes	Stochastic	None

The Deep-RL algorithm developed in this chapter is agnostic to the lane choice model. For simulation purposes, we focus our attention on two models: multiple VOT classes with two routes and stochastic choice (*multiclass binary logit model*) and multiple VOT classes with decision routes and deterministic choice (*multiclass decision route model*). For simulation purposes, we evaluate the utility of a route as the linear combination of the toll and route's travel time, converted to the same units using the VOT for the class (A#9).

4.2.3 Partially observable Markov decision process

MDPs are a discrete time stochastic control process that provide a framework for solving problems that involve sequential decision making [75]. At each time step, the system is in some *state*. The decision maker takes an *action* in that *state*, and the system transitions to the next *state* depending on the transition probabilities, which are only a function of the current *state* and the *action* taken (called the Markov property). Given an *action*, this transition from one *state* to the other generates a reward for each time step and the decision maker seeks to maximize the expected reward across all time steps. Control problems in transportation do not necessarily have the Markov property because of the temporal dependence of congestion pattern. However, by including the simulation time as part of the state, they can be formulated as an MDP.

Partially observable Markov decision processes (POMDPs) are MDPs where the state at any time step is not known with certainty, that is, the state is not fully observable. For

the dynamic pricing problem where a toll operator does not have access to traffic information throughout the network but only at certain locations, POMDPs are a suitable choice. We define the control problem for determining the optimal toll as an POMDP with following components:

- **Timestep**: Tolls are to be optimized over a finite time horizon for each time $k \in \mathscr{T}_\tau$. A finite horizon can represent a morning or an evening peak period on a corridor, or an entire day.

- **State**: We first define $x_c^z(t)$ as the number of vehicles in cell $c \in \mathscr{C}$ belonging to class $z \in Z$ at time $t \in \mathscr{T}$, where $Z = \{(v, d) \mid v \in V, d \in N_d\}$ is the set of all classes, disaggregated by the VOT value and the destination of the vehicle (the origin of a vehicle does not influence lane choice once the vehicle is on the road and is thus ignored). For ML networks where high occupancy vehicles pay a different toll than single/low occupancy vehicles, we can extend Z to include the occupancy level of vehicles, but we leave that analysis for future work (FW#2). The dimensionality of Z impacts the computational performance of the multiclass cell transmission model. Similar to the non-atomic flow assumption commonly used in the transportation literature, we consider $x_c^z(t)$ to be a non-negative real number We denote the state of the POMDP by s comprising of the current toll update step $k \in \mathscr{T}_\tau$ and the values $x_c^z(t_{k\Delta\tau})$ for all cells $c \in \mathscr{C}$ and class $z \in Z$. Thus, the state space S can be written as Equation (4.1). Allowing $\Delta\tau$ to be greater than Δt ($m > 1$) reduces the size of state space compared to choosing $m = 1$, which improves the computational efficiency.

$$S = \{(k, x_c^z(t_{k\Delta\tau})) \mid k \in \mathscr{T}_\tau, c \in \mathscr{C}, z \in Z\} \tag{4.1}$$

- **Observation**: In our model, the observation is done using loop detectors. The detectors measure the total number of vehicles going from one cell to the next and cannot distinguish between vehicles belonging to different classes, so the state is not fully observable. The observation space depends on the location of detectors. We conduct sensitivity analyses with respect to changes in the observation space later in the text. Let

$\boldsymbol{o}(s)$ denote the observation vector for state s and comprise of the measurement of total number of vehicles on each link $(i,j) \in A_{\text{loop}} \subseteq A$ which has a loop detector installed at beginning and end.[1] That is, $\boldsymbol{o}(s) = \{\sum_{z \in Z} \sum_{c \in \mathscr{C}_{(i,j)}} x_c^z(t_{k\Delta\tau}) \mid (i,j) \in A_{\text{loop}}\}$. We assume that we can learn the total number of vehicles on any link by tracking the number of vehicles entering the link (measured at an upstream detector) and the number of vehicles leaving the link (measured at a downstream detector) (A#10). The actual observation is assumed to be Gaussian random variable with the mean as specified and the standard deviation σ_o which models the noise in loop detector measurements. We project negative values of observation, if any, to zero.

- **Action**: Action a in state s is the toll $\beta_{ij}(t_{k\Delta\tau})$ charged for a toll link $(i,j) \in A_{\text{toll}}$, where $\beta_{ij}(\cdot) \in [\beta_{\min}, \beta_{\max}]$. The action is modeled as a continuous variable; the values can be rounded to nearest tenth of a cent or dollar if desired.

- **Transition function**: The transition of the POMDP from a state s to a new state s' given action a, is governed by the traffic flow equations from the CTM model which incorporates the lane choice behavior of travelers. For simulation purposes, we assume that traffic flow throughout the network is deterministic except at diverges where the lane choices of travelers may be stochastic (A#11). We use a multiclass version of the CTM model similar to the model in Chapter 3.

- **Reward**: The reward obtained after taking action a in state s, denoted by $r(s,a)$, depends on the choice of tolling objective. We consider two objectives, revenue maximization and total system travel time (TSTT) minimization, with following definitions of reward:

  – Revenue maximization:

$$r^{\text{RevMax}}(s,a) = \sum_{x=k\Delta\tau}^{(k+1)\Delta\tau-1} \sum_{(i,j) \in A_{\text{toll}}} \left(\beta_{ij}(t_{k\Delta\tau}) \sum_{(h,i) \in A} y_{hij}(t_x) \right), \tag{4.2}$$

  where $y_{hij}(t)$ is the total flow moving from link $(h,i) \in A$ to $(i,j) \in A$ from time

[1] For Figure 4.1, $A_{\text{loop}} = \{(o,a),(a,c),(c,e),(d,f),(g,h),(h,j)\}$.

step t to time step $t + \Delta t$

– Total system travel time minimization:

$$r^{\text{TSTTMin}}(s, a) = -\left(\sum_{x=k\Delta\tau}^{(k+1)\Delta\tau-1} \sum_{c\in\mathscr{C}} \sum_{z\in Z} x_c^z(t_x)\right), \tag{4.3}$$

where the negative sign ensures that reward maximization is equivalent to TSTT minimization.

For the dynamic pricing problem, revenue-maximizing tolls often have a JAH nature where the GPLs are jammed to congestion earlier in the simulation to attract more travelers towards the ML later in the simulation generating more revenue [21, 40]. This undesirable characteristic of optimal policy is also seen in other applications of RL. For example, for ATSC a simpler definition of reward that maximizes amount of flow during a cycle may lead to "evil" optimal policies, where the controller agent holds congestion on the mainline and then gains a larger reward by extending the greens for the main approach [65]. Similarly, Van der Pol and Oliehoek [76] show that with inappropriate definitions of reward, the signal control policy may have unusual flips from green to red.

To overcome the undesired JAH nature, we use reward shaping methods that modify the reward definitions such that the optimal policies have less or no JAH behavior (discussed later in Section 4.4.4). For reward shaping, we quantify the JAH behavior using two statistics defined as a numeric value at the end of simulation. The first statistic, JAH_1, measures the maximum of difference between the number of vehicles in GPLs to the number of vehicles in MLs across all time steps. It is defined as in Equation (4.4), where $A_{\text{GPL}}(A_{\text{ML}})$ are links on the GPL (ML).

$$\text{JAH}_1 = \max_{t\in\mathscr{T}} \left(\sum_{(i,j)\in A_{\text{GPL}}} \sum_{c\in\mathscr{C}_{(i,j)}} \sum_{z\in Z} x_c^z(t) - \sum_{(i,j)\in A_{\text{ML}}} \sum_{c\in\mathscr{C}_{(i,j)}} \sum_{z\in Z} x_c^z(t)\right) \tag{4.4}$$

The value of JAH_1 is dependent on network properties like number of lanes in GPLs and MLs. We also define an alternate statistic JAH_2 that is network independent. We first define $\zeta(t)$, as in Equation (4.5), as the difference between the ratio of current number

of vehicles to the maximum number of vehicles allowed in each cell (corresponding to jam density) for all cells on GPLs with that of MLs.

$$\zeta(t) = \frac{\sum_{(i,j)\in A_{\text{GPL}}} \sum_{c\in\mathscr{C}_{(i,j)}} \sum_{z\in Z} x_c^z(t)}{\sum_{(i,j)\in A_{\text{GPL}}} \sum_{i\in\mathscr{C}_{(i,j)}} l_{ij} k_{\text{jam},ij}} - \frac{\sum_{(i,j)\in A_{\text{ML}}} \sum_{i\in\mathscr{C}_{(i,j)}} \sum_{z\in Z} x_i^z(t)}{\sum_{(i,j)\in A_{\text{ML}}} \sum_{i\in\mathscr{C}_{(i,j)}} l_{ij} k_{\text{jam},ij}} \tag{4.5}$$

JAH_2 can then be defined as a maximum value of $\zeta(t)$ across all time steps, as in Equation (4.6). The value of JAH_2 varies between $[-1, 1]$ with a high positive value indicating more congestion on GPLs before congestion set in the ML.

$$\text{JAH}_2 = \max_{t\in\mathscr{T}} \zeta(t) \tag{4.6}$$

For the given POMDP, a policy $\pi_\theta(a|\boldsymbol{o}(s))$ denotes the probability of taking action a given observation $\boldsymbol{o}(s)$ in state s. We consider stochastic policies parameterized by a vector of real parameters θ. For example, for a policy replaced by a neural network, θ represents the flattened weights and biases for the nodes in the network. Since the action space for the POMDP is continuous, the neural network outputs the mean of the Gaussian distribution of tolls which is then used to sample continuous actions. For simplicity in Deep-RL training, we assume the covariance of the joint distribution of actions to be a diagonal matrix with constant diagonal terms (A#12). Figure 4.3 shows a schematic of the parameterized representation of the policy which takes in the input of observations across the network and returns the mean of the Gaussian toll values for all toll links. MLP stands for multi-layer perceptron which is a feedforward neural network architecture.

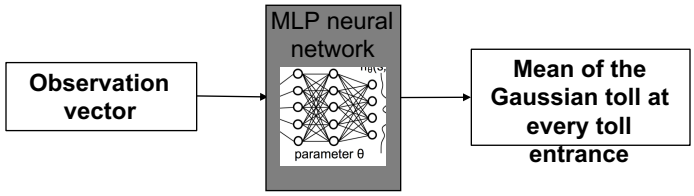


Figure 4.3: Abstract representation of the policy

4.2.4 Episodic reinforcement learning

In an episodic reinforcement learning problem, an agent's experience is broken into episodes, where an episode is a sequence with a finite number of states, actions, and rewards. Since the POMDP introduced in the previous subsection is finite-horizon, the simulation terminates at time $T/\Delta t$. Thus, an episode is formed by a sequence of states, actions, and rewards for each time step $k \in \mathscr{T}_\tau$.

We first define a trajectory $\aleph$ as a sequence of states and actions visited in an episode, that is $\aleph = (s_0, a_0, s_1, a_1, \cdots, s_{|\mathscr{T}_\tau|-1})$, where s_k is same as the state defined earlier indexed by the time k in that state. Let $r(s_k, a_k)$ be denoted by r_k for all $k \in \mathscr{T}_\tau$.

The goal of the RL problem is to find a policy that maximizes the expected reward over the entire episode. The optimization problem can then be written as following:

$$\max_{\pi_\theta(\cdot)} J(\pi_\theta) = \mathbb{E}_\aleph[R(\aleph)|\pi] \tag{4.7}$$

$$R(\aleph) = \sum_{k \in \mathscr{T}_\tau} r_k, \tag{4.8}$$

where, $\mathbb{E}_\aleph[R(\aleph)|\pi] = \int R(\aleph) p_\pi(\aleph) d\aleph$ is the expected reward over all possible trajectories obtained after executing policy π with $p_\pi(\aleph)$ as the probability distribution of trajectories obtained by executing policy π.[2] We do not discount future rewards because tolls are optimized over a short time period (like a day or a morning/evening peak).

We define a few additional terms used later in the text. Let $V^\pi(s_k) = \mathbb{E}_\aleph \sum_{k'=k}^{|\mathscr{T}_\tau|} r_{k'}$ be the value function which evaluates the expected reward obtained from state s_k till the end of episode following policy π. Similarly, we define the Q-function, denoted by $Q^\pi(s_k, a_k)$, as the expected reward obtained till the end of episode from state s_k after taking action a_k and following policy π thereafter. Last, the advantage function $A^\pi(s_k, a_k) = Q^\pi(s_k, a_k) - V^\pi(s_k)$, defined as the difference between Q-function and value function, determines how much better or worse is an action than other actions on average, given the current policy.

[2]Defining an expectation conditioned over a function (π) instead of a random variable is a slight abuse of notation, but is commonly used in the RL literature.

The solution of this POMDP is a vector θ^* that determines the policy which optimizes the objective under certain constraints on the policy space. Commonly considered policy constraints for the dynamic pricing of MLs include the following:

1. Tolls levied for a longer distance are higher than tolls levied for a shorter distance from the same entrance: with the choice of tolling structure (assumption A#6) where tolls are charged at every diverge, this constraint is already satisfied.

2. The ML is always operated at a speed higher than the minimum speed limit (called the speed-limit constraint): in our model, we allow violation of this constraint on the ML. We observe that, given the stochasticity in lane choice of travelers and demand, bottlenecks can occur at merges and diverges which can result in an inevitable spillover on managed lanes during congested cases. Thus, a hard constraint keeping the ML congestion free throughout the learning period is not useful. We instead quantify the violation of the speed-limit constraint using the time-space diagram of the cells on the ML. We define `%-violation` as the proportion of cell-timestep pairs on the time-space diagram where the speed limit constraint is violated, expressed as percentage. Mathematically,

$$\texttt{\%-violation} = \frac{\sum_{(i,j)\in A_{\text{ML}}} \sum_{c\in\mathscr{C}_{(i,j)}} \sum_{t\in\mathscr{T}} I_c^t}{|\mathscr{T}| \sum_{(i,j)\in A_{\text{ML}}} |\mathscr{C}_{(i,j)}|} \times 100, \tag{4.9}$$

   where, I_c^t is an indicator variable which is 1 if the number of vehicles in the cell c in time step t is higher than the desired number of vehicles in the cell and 0 otherwise. The desired number of vehicles in each cell is determined from the density corresponding to the minimum speed limit on the fundamental diagram. As discussed in Section 4.4, allowing the speed-limit constraint to be violated in our model is not restrictive as the best-found policies for each objective have `%-violation` values of less than 2% for all networks tested.

3. Toll variation from one time step to the next is restricted: we do not explicitly model this constraint. If the tolling horizon is "sufficiently" large (say 5 minutes), a large change in tolls from one toll update to the next can be less of a problem. In our

experiments, the optimal tolls are structured and do not oscillate significantly.

4. Tolls are upper and lower bounded by a value: we model this by clipping the toll output by the function approximator within the desired range $[\beta_{\min}, \beta_{\max}]$.

Next, we discuss the solution methods used to solve the POMDP using Deep-RL methods and other heuristics.

4.3 Solution methods

4.3.1 Deep reinforcement learning algorithms

Deep reinforcement learning algorithms can be broadly categorized into value-based methods and policy-based methods. The former methods try to learn the value functions and use approaches based on dynamic programming to solve the problem, while the latter methods try to learn the policy directly based on the observations. Policy gradient methods work well with continuous state and action spaces, making it a preferred choice for the toll optimization problem.

Derivative-free optimization and gradient-based optimization are two types of policy-based methods. We focus on the methods relying on derivatives as they are considered to be data efficient [77]. Providing an overview of the state-of-the-art of policy gradient methods to solve RL problems is out of the scope of this work. We refer the reader to Schulman [77] for additional details. In this chapter, we choose two of the commonly used algorithms for solving the problem: the vanilla policy gradient (VPG) algorithm and the proximal policy optimization (PPO) method from Schulman et al. [78], which we describe next.

The algorithms use the derivative of the objective function with respect to the policy parameters to improve them using stochastic gradient descent. The methods differ in calculation of the derivatives and the update of parameter θ. We can express the derivative of

$J(\pi_\theta)$ with respect to θ as:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \mathbb{E}_{\aleph}[R(\aleph)|\pi] \tag{4.10a}$$

$$= \nabla_\theta \int_{\aleph} P(\aleph|\theta) R(\aleph) d\aleph \tag{4.10b}$$

$$= \int_{\aleph} \nabla_\theta P(\aleph|\theta) R(\aleph) d\aleph \tag{4.10c}$$

$$= \int_{\aleph} P(\aleph|\theta) \nabla_\theta \log P(\aleph|\theta) R(\aleph) d\aleph \qquad \left(\text{since } \nabla_\theta \log P(\aleph|\theta) = \frac{1}{P(\aleph|\theta)} \nabla_\theta P(\aleph|\theta)\right) \tag{4.10d}$$

$$= \mathbb{E}_{\aleph} \left[\nabla_\theta \log P(\aleph|\theta) R(\aleph)\right] \tag{4.10e}$$

$$= \mathbb{E}_{\aleph} \left[\sum_{k=0}^{|\mathscr{T}_\tau|} \nabla_\theta \log(\pi_\theta(a_k|s_k)) R(\aleph)\right], \tag{4.10f}$$

where we first convert the probability of a trajectory into a product of the probabilities of taking certain actions in each state, and then convert this product into a sum. As a result, the derivative in the RHS of Equation (4.10f) can be easily obtained by performing back propagation on the policy neural network.

The expectation in Equation (4.10f) can be approximated by averaging over a finite number of trajectories. Let $\mathscr{N} = \{\aleph_i \mid i \in 1, 2, ...\}$ be the set of trajectories obtained using policy $\pi_\theta(\cdot)$. Then, we can write:

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{|\mathscr{N}|} \sum_{\aleph \in \mathscr{N}} \left[\sum_{k=0}^{|\mathscr{T}_\tau|} \nabla_\theta \log(\pi_\theta(a_k|s_k)) R(\aleph)\right]. \tag{4.11}$$

In the above formulation the likelihood of actions taken along the trajectory is affected by reward over entire trajectory. However, it is more intuitive for an action to influence the reward obtained only after the time step when it was implemented. It can be shown that the right hand side of the expression in Equation (4.11) is equivalent to the following expression:

$$\frac{1}{|\mathscr{N}|} \sum_{\aleph \in \mathscr{N}} \left[\sum_{k=0}^{|\mathscr{T}_\tau|} \nabla_\theta \log(\pi_\theta(a_k|s_k)) \hat{R}(k)\right], \tag{4.12}$$

where $\hat{R}(k)$ is the reward-to-go function at time k, given by $\hat{R}(k) = \sum_{k'=k}^{|\mathscr{T}_\tau|} r_{k'}$. This new expression for the gradient of the objective requires sampling of fewer trajectories and generates a low-variance sample estimate of the gradient.

Additionally, the variance can be further reduced by using the advantage function estimates instead of reward-to-go function [79]. VPG uses the following form for approximating the derivative:

$$\nabla_\theta J(\pi(\theta)) \approx \frac{1}{|\mathscr{N}|} \sum_{\aleph \in \mathscr{N}} \left[\sum_{k=0}^{|\mathscr{T}_\tau|} \nabla_\theta \log(\pi_\theta(a_k|s_k)) \hat{A}_k \right], \tag{4.13}$$

where $\hat{A}_k$ is the estimate of advantage function, $A^{\pi_\theta}(s_k, a_k)$, from current time k till the end of episode, following the policy from which the given trajectory is sampled. We use the generalized advantage estimation (GAE) technique to estimate $\hat{A}_k$ which requires an estimate of the value function [79]. We use value function approximation to estimate of $V^\pi(s_k)$ using a neural network as the functional approximator. Let $\hat{V}_\phi(s_k)$ denote the estimate of $V^\pi(s_k)$, parameterized by a real vector of parameters ϕ. The algorithm starts with an estimate of ϕ (ϕ_0) and iteratively improves it by minimizing the squared difference with the reward-to-go value from the trajectory. The update in ϕ parameters are evaluated using Equation (4.14):

$$\phi_{n+1} = \underset{\phi}{\operatorname{argmin}} \frac{1}{|\mathscr{N}||\mathscr{T}_\tau|} \sum_{\aleph \in \mathscr{N}} \sum_{k=0}^{|\mathscr{T}_\tau|} \left(V_\phi(s_k) - \hat{R}(k) \right)^2. \tag{4.14}$$

More details on GAE are provided in Schulman et al. [79].

VPG updates the value of θ parameter from iteration n to $n+1$ using the standard gradient ascent formula:

$$\theta_{n+1} = \theta_n + \alpha \nabla_\theta J(\pi(\theta_n)). \tag{4.15}$$

In Equation (4.15), an inappropriate choice of the learning rate α can lead to large policy updates from one iteration to the next which can cause the objective values to fluctuate. The PPO algorithm modifies the policy update to take the biggest possible improvement using the data generated from current policy while ensuring improvement in the objective.

It performs specialized clipping to discourage large changes in the policy. The policy update for PPO is given by:

$$\theta_{n+1} = \underset{\theta}{\operatorname{argmax}} \frac{1}{|\mathscr{N}|} \sum_{\aleph \in \mathscr{N}} \left[\sum_{k=0}^{|\mathscr{T}_\tau|} \min \left(r_k(\theta) \hat{A}^{\pi_{\theta_n}}(s_k, a_k), \texttt{clip}(r_k(\theta), 1-\epsilon, 1+\epsilon) \hat{A}^{\pi_{\theta_n}}(s_k, a_k) \right) \right], \tag{4.16}$$

where $r_k(\theta)$ is the ratio of probabilities following a policy and the policy in the current iteration (θ_n) given by Equation (4.17), and the $\texttt{clip}(\cdot)$ function, given by Equation (4.18), restricts the value of first argument between the next two arguments.

$$r_k(\theta) = \frac{\pi_\theta(a_k|s_k)}{\pi_{\theta_n}(a_k|s_k)} \tag{4.17}$$

$$\texttt{clip}(r, 1-\epsilon, 1+\epsilon) = \begin{cases} 1-\epsilon, & \text{if } r \leq 1-\epsilon \\ r, & \text{if } 1-\epsilon < r < 1+\epsilon \\ 1+\epsilon, & \text{if } r \geq 1+\epsilon. \end{cases} \tag{4.18}$$

The clipping operation selects the policy parameters in the next iteration such that the ratio of action probabilities in iteration $n+1$ to iteration n are between $[1-\epsilon, 1+\epsilon]$, where ϵ is a small parameter, typically 0.01. Policy updates for PPO can be solved using the Adam gradient ascent algorithm, a variant of stochastic gradient ascent with adaptive learning rates for different parameters [78, 80].

The structure for both algorithms is presented in Algorithm 6. For the experiments, we develop a new RL environment for macroscopic simulation of traffic similar to the current RL benchmarks (called "gym" environments) and customize the open-source implementation of both algorithms provided by OpenAI Spinningup [81] to work with our new environment.

Algorithm 6 Policy gradient algorithm for dynamic pricing [81]

Input: initialize policy parameters θ_0 and value function parameters ϕ_0
for **do** $n = 0, 1, 2, \cdots$
Collect set of trajectories $\mathscr{N}_n = \{\aleph_n\}$ by running policy $\pi_n = \pi_{\theta_n}$ in the environment
Compute rewards to go $\hat{R}_k$
Compute advantage estimates using rewards-to-go and generalized advantage estimation
Update policy parameters using either VPG or PPO update:

- **VPG**: Estimate policy gradients using Equation (4.13) and update policy parameters using Equation (4.15), or
- **PPO**: Update policy parameters by solving Equation (4.16) using Adam gradient ascent algorithm

Update value function approximation parameter (used for advantage estimation) in Equation (4.14) using Adam gradient descent
end for

4.3.2 Feedback control heuristic

We compare the performance of Deep-RL algorithms against a feedback control heuristic based on the measurement of total number of vehicles in the links on ML. We customize the `Density` heuristic in the previous chapter to charge varying tolls for different toll links.

Define $\text{ML}(i, j)$ as the set of links on the ML used by a traveler upon first entering the ML using the toll link $(i, j) \in A_{\text{toll}}$ until the next merge or diverge. For the network in Figure 4.1, $\text{ML}(a, b) = \{(b, d)\}$, $\text{ML}(c, d) = \{(d, f)\}$, $\text{ML}(f, i) = \{(f, i)\}$, and $\text{ML}(h, i) = \{(i, k)\}$. This definition allows the sets $\text{ML}(i, j)$ to be mutually exclusive and exhaustive in the space of all links on the ML. That is,

$$\text{ML}(i,j) \cap \text{ML}(k,l) = \Phi \qquad \forall\ (i,j) \in A_{\text{toll}}, (k,l) \in A_{\text{toll}}, (i,j) \neq (k,l)$$
$$\bigcup_{(i,j) \in A_{\text{toll}}} \text{ML}(i,j) = A_{\text{ML}}.$$

We assume that the feedback control heuristic updates the tolls for each toll link $(i, j) \in A_{\text{toll}}$ based on the density observations on links in $\text{ML}(i, j)$, that is, detectors are installed on each link in the ML and only those detectors are used to update the toll (A#13). The toll value for an update time $(k + 1) \in \mathscr{T}_\tau$ is based on the toll value in the previous

update step adjusted by the difference between the desired and current numbers of vehicles. The toll update is given by Equation (4.19),

$$\beta_{ij}(t_{(k+1)\Delta\tau}) = \beta_{ij}(t_{k\Delta\tau}) + P \times \left(X_{\mathrm{ML}(i,j)}(k) - X^{\mathrm{desired}}_{\mathrm{ML}(i,j)}\right), \tag{4.19}$$

where $X_{\mathrm{ML}(i,j)}(k)$ is the total number of vehicles on links in $\mathrm{ML}(i,j)$ before updating tolls at time $k+1$ and $X^{\mathrm{desired}}_{\mathrm{ML}(i,j)}$ be the desired value of the number of vehicles on the links in $\mathrm{ML}(i,j)$. P is the regulator parameter, with units \$/veh, controlling the influence of difference between the desired and current number of vehicles on the toll update. A typical desired value is the number of vehicles corresponding to the critical density on the ML link. We generalize the desired number of vehicles by defining $X^{\mathrm{desired}}_{\mathrm{ML}(i,j)}$ as:

$$X^{\mathrm{desired}}_{\mathrm{ML}(i,j)} = \sum_{(g,h)\in \mathrm{ML}(i,j)} \eta k_{\mathrm{critical},(g,h)} l_{gh}, \tag{4.20}$$

where, $k_{\mathrm{critical},(g,h)}$ is the critical density for link $(g,h) \in A$ and η is the scaling parameter varying between $(0,1]$ that sets the desired number of vehicles to a proportion value of the number of vehicles at critical density. We calibrate the feedback control heuristic for different values of desired density and regulator parameter. In principle, both η and P can vary with time and the toll location; however, determining the "optimal" variability in these parameters is a control problem in itself, exploring which is left as part of the future work (FW#3).

In Section 4.4.5 we also compare the performance of algorithms making the *full observability* assumption against the Deep-RL algorithms which do not make that assumption. We choose two algorithms from our previous work: the algorithm based on value function approximation (VFA) using look-up tables [21], and the multiagent reinforcement learning algorithm that learns value functions separately for each toll gantry (SparseV algorithm) [63]. Comparing the performance of Deep-RL methods against the hybrid MPC method in Tan and Gao [12] requires extensive analysis and will be a part of the future work (FW#4).

4.4 Experimental analysis

4.4.1 Preliminaries

We conduct our analysis on four different networks. The first is a network with single entrance and single exit (SESE) commonly used in the managed lane pricing literature. The next two are the double entrance single exit (DESE) network and the network for toll segment 2 of the LBJ TEXpress lanes in Dallas, TX (LBJ). The DESE network includes two toll locations for modeling *en route* lane changes. The LBJ network has four toll locations. Last is the network of the northbound Loop 1 (MoPac) Express lanes in Austin, TX. The MoPac network has three entry locations to the MLs and two exit locations.

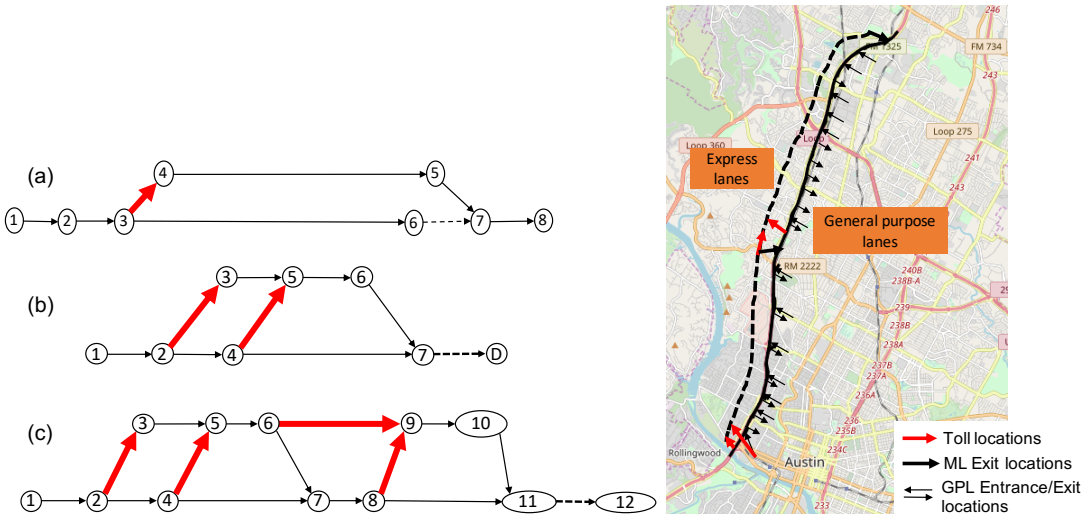
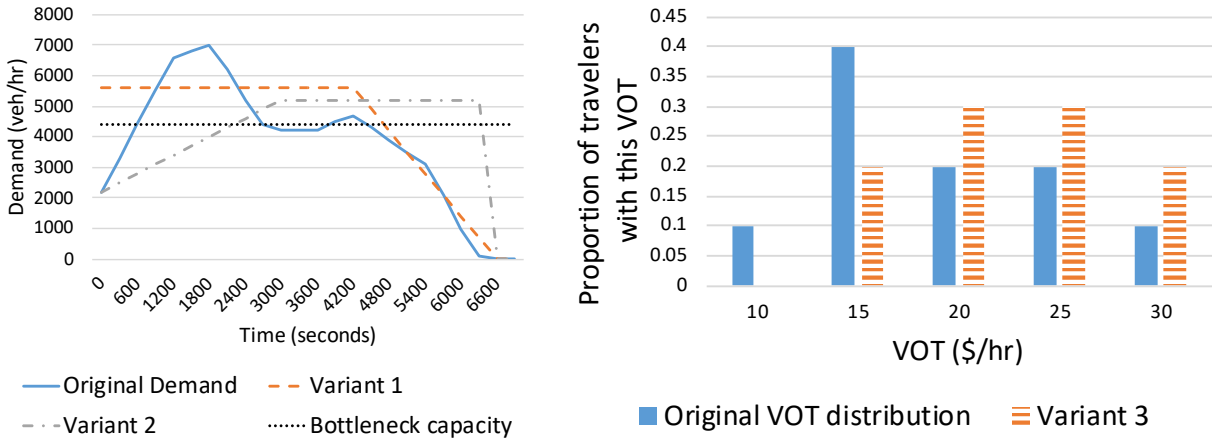


Figure 4.4: Abstract representation of (a) single entrance single exit (SESE) network, (b) double entrance single exit (DESE) network, (c) LBJ network, and (d) Northbound MoPac express lane network (latitude-longitude locations of MLs are shifted to the left to show the locations of toll points and exits from the managed lane). The tolls are collected on the links with higher thickness.

Figure 4.4 shows the networks, where the thick lines denote the links where tolls are collected. The demand distribution for the first three networks is artificially generated and follows a two-peak pattern (refer to the original demand curve in Figure 4.5a), while the demand for the MoPac network is derived from a dynamic traffic assignment model of the Travis County region. There are a total of 105 origin-destination pairs in the MoPac network with a total demand of 49,273 vehicles using the network in three hours of the evening peak.



(a) (b)

Figure 4.5: (a) Demand distributions used for the SESE, DESE and LBJ networks and its variants, and (b) VOT distribution and its variant

Table 4.2 shows the values of parameters used for different networks. Five VOT classes were selected for each network and the same VOT distribution was used. Figure 4.5b shows this VOT distribution (labelled "original"; in some experiments we vary this distribution.)

Table 4.2: Values of parameters used in the simulation

	SESE	DESE	LBJ	MoPac		Parameter	Value
Corridor length (miles)	7.3	1.59	2.91	11.1		$\beta_{\min}$	\$0.1
Simulation duration (hour)	2	2	2	3		$\beta_{\max}$	\$4.0
$\Delta\tau$ (seconds)	60	300	300	300		q_{ij} (vphpl)	2200
ν_{ij} (mph)	55	55	55	65		$k_{\text{jam},ij}$ (veh/mile)	265
σ_o (veh/hr)	50	50	50	50		ν_{ij}/w_{ij}	3
σ_d (veh/hr)	10	0	0	100		Δt (seconds)	6

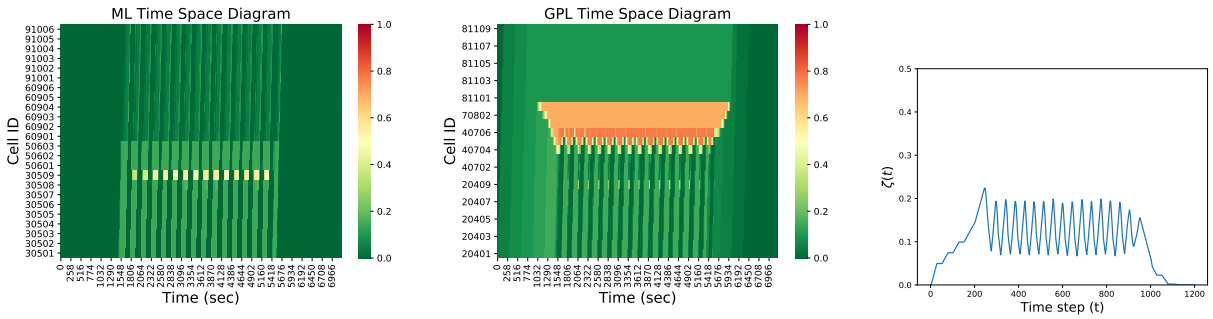
A feedforward multilayer perceptron was selected as the neural network. Hyperparameter tuning was conducted, and the architecture with two hidden layers and 64 nodes in each layer was selected. For the MoPac network, three hidden layers with 128 nodes each were selected. The values of other hyperparameters for Deep-RL training are as follows: learning rate for policy update equals 10^{-4}, learning rate for value function updates is 10^{-3}, number of iterations for value function updates is 80, and the γ^{GAE} and λ^{GAE} values for the GAE method are 0.99 and 0.97, respectively. Each network was simulated for a number of

iterations ranging between 100 and 200 where the average in each iteration was reported over 10 episodes.

4.4.2 Validating JAH statistics

In this subsection, we discuss how the JAH statistics defined in Equations 4.4 and 4.6 are meaningful in capturing the jam-and-harvest nature of the revenue maximizing profiles. We simulate random toll profiles on the LBJ network and record the congestion profiles for two values of JAH_2: 0.22 and 0.49.[3]

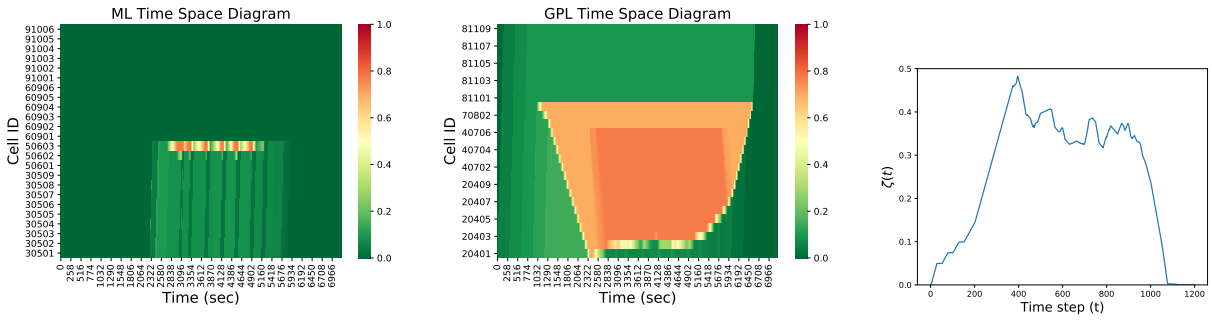
Figures 4.6 and 4.7 show the plots for the time space diagram on managed lane and general purpose lane, and the variation of $\zeta(t)$ for two different toll profiles leading to JAH_2 values of 0.22 and 0.49, respectively. The scale on the time-space diagrams varies from 0, representing no vehicles, to 1, representing jam density. The cell id value on the y-axis is a six-digit number where the first two digits are the tail node of the link, the second two digits are the head node of the link, and the last two digits are the index of the cell number on the link starting from index 1 for the first cell near the tail node. Thus, the increasing value of cell IDs on the y-axis indicates the downstream direction.



(a) (b) (c)

Figure 4.6: Plots for $JAH_2 = 0.22$

[3]The JAH_2 values varied between 0.2 and 0.5 for this network as shown in Figure 4.11



(a) (b) (c)

Figure 4.7: Plots for $JAH_2 = 0.49$

As observed, higher value of JAH statistics results in higher congestion on the GPL relative to the ML. When $JAH_2 = 0.22$, vehicles use the ML starting from 1500 seconds into the simulation. Whereas, when $JAH_2 = 0.49$, vehicles do not enter the managed lane until approximately 2300 seconds into the simulation, by which the GPLs are heavily congested, indicating more jam-and-harvest behavior .

Table 4.3 shows values of revenue, TSTT, and JAH_1 for the two toll profiles simulated. We see that the JAH_1 statistic is also high when the JAH_2 statistic is high. The highest revenue is obtained for the highest value of JAH_2 value. TSTT values follow the reverse trend as the revenue: high JAH statistic leads to low TSTT. These experiments help quantify the abstract "jam-and-harvest" nature used in the literature. In Section 4.4.4, we use reward shaping to generate toll profiles with low JAH_i values ($i = \{1, 2\}$).

Table 4.3: Value of different statistics for different cases

Figure	Revenue ($)	TSTT (hr)	JAH_1 (vehicles)	JAH_2
Figure 4.6	1203.68	1018.7	451.73	0.22
Figure 4.7	4106.03	1421.05	997.23	0.49

4.4.3 Learning performance of Deep-RL

Learning for different objectives

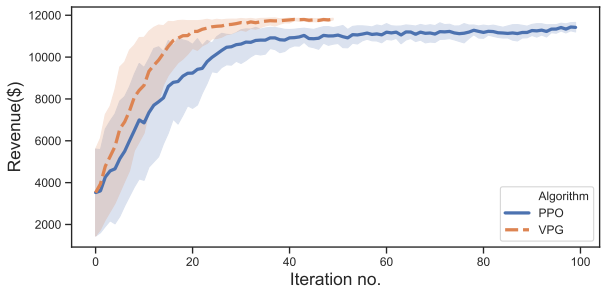
We next compare the learning performance of the VPG and PPO Deep-RL algorithms for both revenue maximization and TSTT minimization objectives. Figure 4.8 show the plots of variation of learning for two objectives for all four networks over 200 iterations. The

average in each iteration is reported over 10 random seeds, and for each random seed 10 trajectories are simulated to perform policy updates in Equations (4.15) and (4.16).

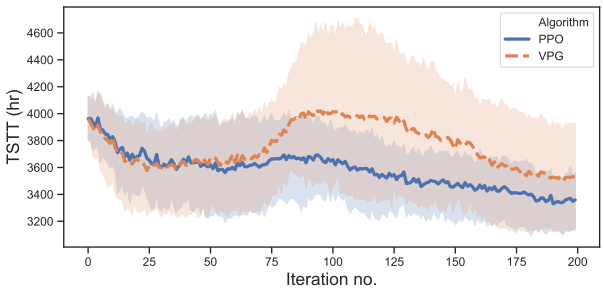
We make the following observations. First, both Deep-RL algorithms are able to learn "good" objective values within 200 iterations, evident in the increasing trend of the average revenue for the revenue maximization objective and a decreasing trend of the average TSTT for the TSTT minimization objective. For the revenue maximization objective, the average revenue values converge to a high value for all networks. For the TSTT minimization objective, the average TSTT values for SESE (Figure 4.8b) and DESE (Figure 4.8d) networks do not converge; however a decreasing trend is evident. The VPG algorithm for the DESE network in Figure 4.8d shows divergence towards the end. This behavior can be attributed to the lack of convergence guarantees for gradient-based algorithms in stochastic settings, where the algorithms may converge to a local optimum or may not converge within desired number of iterations. Therefore, we recommend tracking the value of policy parameters (θ) that achieve the best-found objective over iterations.

We argue that learning for the revenue maximization objective is easier than learning for the TSTT minimization objective. This is because the reward definition for revenue maximization in Equation (4.2) involves the action values (in terms of $\beta_{ij}(\cdot)$) and thus incorporates a direct feedback on the efficiency of current toll. On the other hand, for the TSTT minimization objective, Equation (4.3) does not incorporate the toll values directly and the feedback on current toll is only obtained at the end of simulation when the TSTT value is generated. This is known as the *credit assignment problem* in the RL literature where it is unclear which actions over the entire episode were helpful. The credit assignment problem can potentially be addressed by reframing the reward definition for the TSTT minimization objective, but this analysis is left as part of the future work (FW#5).

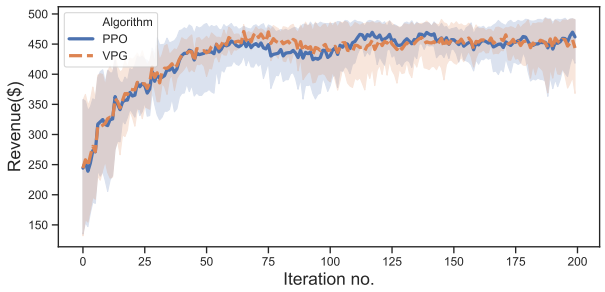
Second, we observe that there is no evident difference in the performance of VPG and PPO algorithms. For the revenue maximization objectives, the algorithms perform "almost identically" with values of average revenue of PPO within 5% of the average revenue values of VPG algorithm at any iteration. For the TSTT minimization objective, we observe that PPO prevents high variation in average TSTT values from one iteration to the next, whereas the VPG algorithm shows higher oscillations (evident in Figures 4.8b and 4.8d). The variance



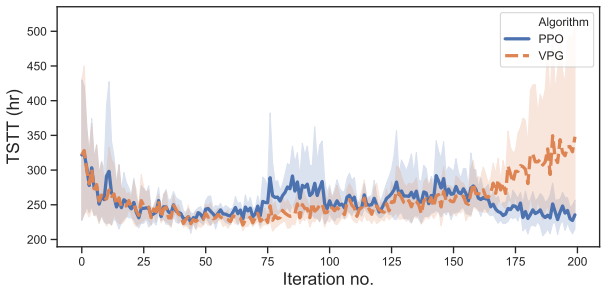
(a) SESE Revenue Maximization.



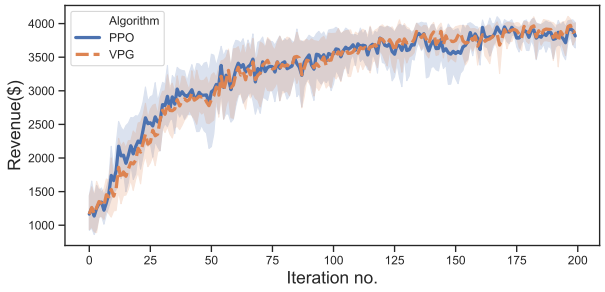
(b) SESE TSTT Minimization.



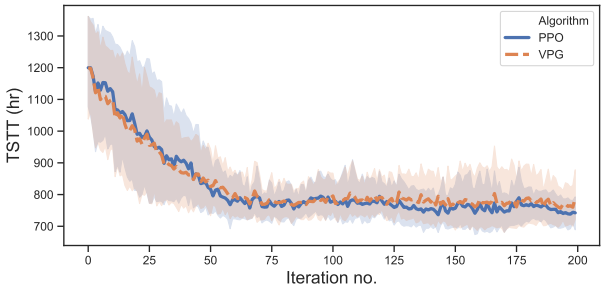
(c) DESE Revenue Maximization.



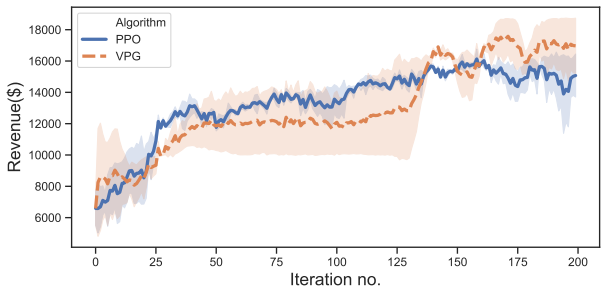
(d) DESE TSTT Minimization.



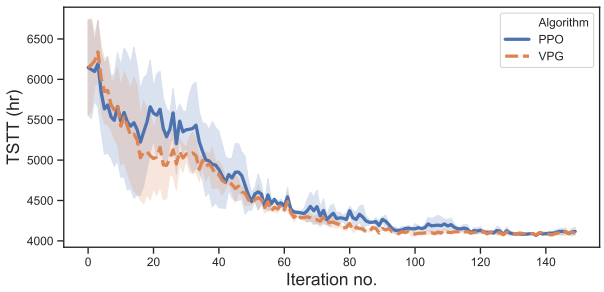
(e) LBJ Revenue Maximization.



(f) LBJ TSTT Minimization.



(g) MoPac Revenue Maximization.



(h) MoPac TSTT Minimization.

Figure 4.8: Plot of average objective value and the confidence interval with iteration over 10 random seeds for the four networks

in the average TSTT values is also higher for the VPG algorithm for the TSTT minimization objective.

Last, in contrast to our expectation that a larger network with high dimensional action space might require large number of iterations to converge, we observe that for both LBJ and MoPac networks, the average objectives converge within 200 iterations, which is equivalent to simulating 2000 episodes with 2000×2 hours/5 minutes = 48000 action interactions with the environment. Both networks mimic the real-world implementations of managed lanes, and thus we argue that learning is possible within a reasonable number of interactions with the environment even for real-world networks. The amount of data required for training Deep-RL models is often considered its major limitation [64]; however, for the dynamic pricing problem it is not a constraining factor.

Next, we report the computation time needed for training the networks in Table 4.4. The run times are reported on a Unix machine with 8 GB RAM and are computed starting when the algorithms begin execution till the end of desired number of iterations. As observed, both Deep-RL algorithms show minor to no difference. The total computation time for training of algorithm for an objective is less than half a hour for the first three networks. For the MoPac network, the computation time is around 23 hours. The computational bottleneck is the traffic flow simulation using multiclass cell transmission model. For the MoPac network there are $|Z| = 65$ classes and $|\mathscr{C}| = 258$ cells, and thus updating $65 \times 258 = 16{,}770$ flow variables for every time step is time consuming. Efficient implementation of CTM model with parallel computations can help improve the efficiency of training. We note that the 23.39 hours spent for training are conducted offline on a simulation model. Once the model is trained, it can be transferred with less effort to real-world settings. We conduct tests on transferability of learned algorithms to new domains in Section 4.4.3.

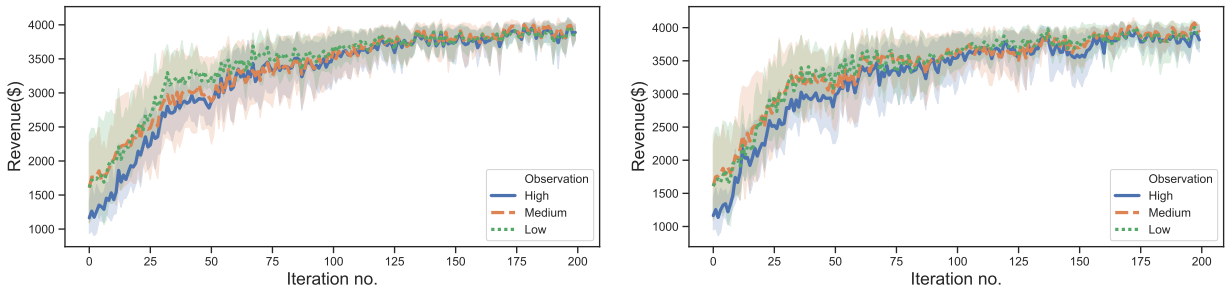
Table 4.4: Computation time for Deep-RL training

Network	Computation time per iteration for simulating 10 episodes (seconds)		Total average computation time for training (hours)
	VPG	PPO	
SESE	7.00	6.99	0.39
DESE	3.59	3.57	0.20
LBJ	7.51	7.49	0.42
MoPac	420.99	419.2	23.39

Impact of observation space

We also test the impact of observation space on the learning of Deep-RL algorithms. For the LBJ network, the results in Figures 4.8e and 4.8f assumed that flows are observed on all links (which we term `High` observation). We consider two additional observation cases: (a) observing links $(3,5),(4,7),(6,9)$, and $(8,11)$ (`Medium` observation), and (b) only observing link $(6,9)$ in the network (`Low` observation). Figure 4.9 shows the learning results for revenue maximization objectives for the two algorithms for three levels of observation space.

We observe that changing the observation space has a minor impact on learning rate. This result was unexpected, and suggests that good performance can be obtained with relatively few sensors. We speculate that this happens due to the spatial correlation of the congestion pattern on a corridor (where observing additional links does not add a new information for setting the tolls). The computation time differences on using different observation spaces were also not significant.



(a) (b)

Figure 4.9: Plot of the average revenue with iteration over 5 random seeds for the three levels of observation for (a) VPG algorithm, and (b) PPO algorithm for the LBJ network

These findings indicate that a toll operator can learn toll profiles optimizing an objective without placing sensors on all links, which is a lower cost alternative than observing all links. Future work will be devoted to the cost-benefit analysis of different sensor-location combinations assuming variability in sensing errors across different sensors. (FW#6)

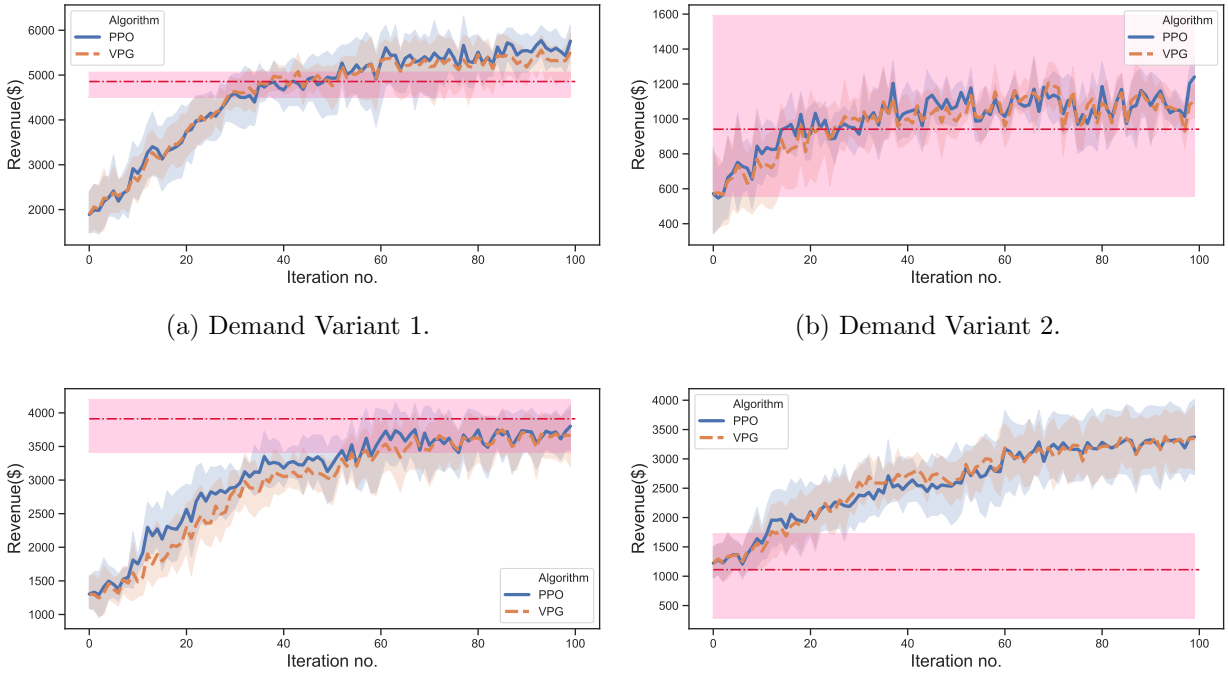
Learning for varied inputs and transferability analysis

In this section, we consider how Deep-RL algorithms perform for varied set of inputs and how the policies trained on one set of inputs perform when transferred to new inputs without retraining for the new inputs. This analysis is useful for a toll operator who trains the algorithm in a simulation environment for certain assumptions of input. For the policy to transfer, the observation space in the new setting must be identical to the setting where the transferred policy is trained. We only consider cases for changes in input demand distribution, VOT distribution, and lane choice model. Transferability of Deep-RL algorithms trained on one network to other networks or the same network with new origins and destinations requires extensive investigation and is a topic for future research (FW#7).

We consider the revenue-maximizing policy for the LBJ network and consider four different input cases. The first two cases consider new demand distributions (Variant 1 and Variant 2) shown in Figure 4.5a. The third case considers a new VOT distribution (Variant 3) shown in Figure 4.5b. And, the last case uses a multiclass binary logit model with scaling parameter 6 for modeling driver lane choice [74]. For each case, we also directly apply the policy obtained at the final iteration of training on the LBJ network for the revenue-maximization objective with the original demand, VOT distribution, and lane choice model (Figure 4.8e).

Figure 4.10 show the plots of variation of revenue with iterations while learning from scratch for both VPG and PPO algorithms and the average revenue (and its full range of variation) obtained from the transferred policy for the new inputs. The average is reported over 100 runs of the transferred policy for new inputs without retraining.

First, we observe that learning for the new input configurations "converges" within 100 iterations for all four cases. This observation indicates the Deep-RL algorithms can iteratively learn "good" toll profiles regardless of the input distribution. This is a significant advantage over the MPC-based algorithms in the literature that require assumptions on



(a) Demand Variant 1.

(b) Demand Variant 2.

(c) VOT Variant 3.

(d) Stochastic lane choice.

Figure 4.10: Comparing learning-from-scratch performance of the VPG and PPO algorithms on different input distributions with the policy transferred after learning on the original distribution (shown as a horizontal line-dot pattern) for the LBJ network

driver behavior and inputs to solve the optimization problem at each time step. Similar to the previous cases, both VPG and PPO algorithms perform almost identically with less than $\sim 10\%$ difference in the objective values at any iteration for the four cases. This is in contrast to the other environments used for testing Deep-RL algorithms like Atari games and `MuJoCo` where the PPO algorithm is significantly better than the VPG algorithm [78]. This is because the state update in the ML pricing problem is not drastically influenced by the toll actions, unlike the high uncertainty in the state transition in the Atari and `MuJoCo` environments. Thus, the VPG algorithm does not produce large-policy updates and has no relative disadvantage over the PPO algorithm, explaining their almost-identical performance.

Second, the average revenue of the transferred policy is within $5-12\%$ of the average revenue at termination while learning from scratch. For case 3 with VOT variant, the transferred policy does even better than the policy learned from scratch after 100 iterations of training. The observations from the first three cases suggest that even though the Deep-RL

algorithms were not trained for the new inputs, they are able to learn characteristics of the congestion in the network and perform well (on an average) on the new inputs. However, for case 2, the transferred policy has a lot of variance in the generated revenue; this is because small changes in input tolls have higher impact on generated revenue for demand Variant 2.

Third, contrary to the first three cases, the transfer of policy in case 4 did not work well: the average revenue of transferred policy is 40% of the maximum revenue obtained. This is because the multiclass logit model predicts significantly different proportion of splits of travelers at a diverge and thus have a significant impact on the evolution of congestion. Both cases 3 and 4 impact the split of travelers at the diverge, yet the performance of transferred policy is very different for both cases. This finding suggests that the driver lane choice model should be carefully selected and calibrated for Deep-RL training for reliable transfer to the real-world environments, whereas the demand and VOT distributions are less important.

4.4.4 Multi-objective optimization

We next focus our attention on multiple optimization objectives together. In the literature, revenue maximization and TSTT minimization objectives are shown to be conflicting [21], that is toll policies generating high revenue have a high value of TSTT. Finding toll profiles that satisfy both objectives to a degree is the focus of this section.

We consider how different objectives vary with respect to each other for 1000 randomized toll profiles simulated for all four networks. Figure 4.11 shows the plots of variation of TSTT, JAH_1, JAH_2, `%-violation`, and the total number of vehicles exiting the system (throughput) against the revenue obtained from the toll policies. The figure also shows the values of objectives from the toll profiles generated by Deep-RL algorithms where "DRLRevMax" indicates toll profiles from the revenue maximization objectives and "DRLTSTTMin" indicates toll profiles from the TSTT minimization objective.

We make following observations:

1. First, the best toll profiles generated from Deep-RL algorithm are the best found among the other randomly generated profiles for the respective objectives (except for TSTT minimization on SESE network where the Deep-RL algorithm did not converge after

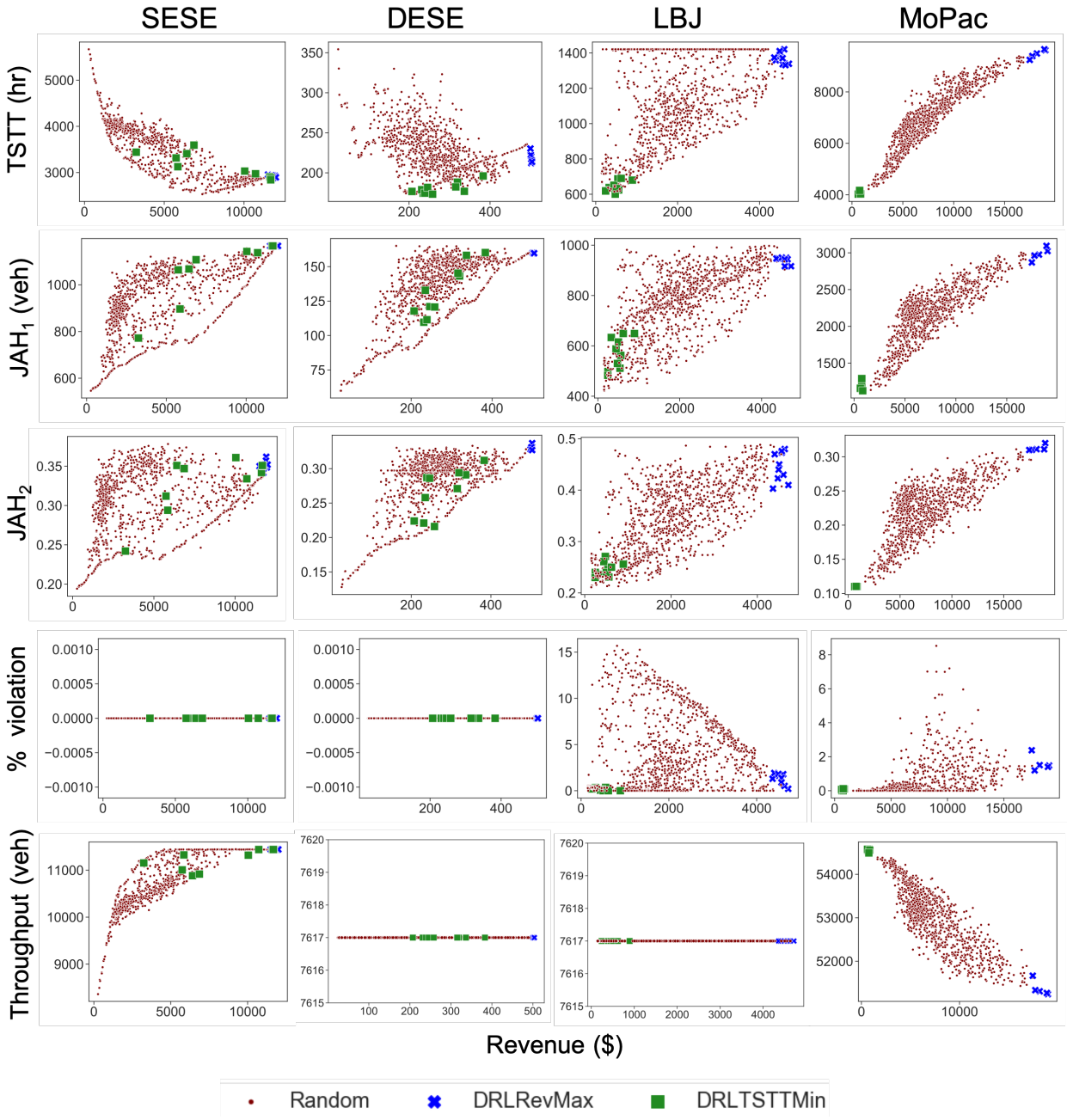


Figure 4.11: Plot of various objectives against the revenue for 1000 randomly generated toll profiles (Random) and the profiles generated from Deep-RL for revenue maximization (DRLRevMax) and TSTT minimization (DRLTSTTMin) objectives

200 iterations).

2. Second, similar to the trends in the literature, toll profiles generating high revenue also generate high values of TSTT for the LBJ and MoPac networks. However, for the SESE and DESE networks, the trend does not hold. This behavior, where revenue-maximizing tolls do not differ significantly from the TSTT-minimizing tolls is possible for networks where GPLs are jammed quickly enough. Once the GPL is jammed, revenue is maximized by charging the highest possible toll while sending maximum number of vehicles towards the ML. Such tolls will also generate low values of TSTT as they utilize the ML to its full capacity from that time step onwards. This finding indicates that, depending on the network properties and the inputs, the two objectives may not always be in conflict with each other. We leave a detailed analysis of how different network characteristics impact the differences between revenue-maximizing and TSTT-minimizing tolls for future work (FW#8).

3. Third, we see that tolls generating high revenue also have high values of JAH_1 and JAH_2 statistics. The tolls generating low TSTT, however, do not have a fixed trend. For example, for the MoPac network, tolls generating low TSTT have lower revenue and thus have lower values of JAH statistics; however, for the other networks, JAH statistics are also relatively high for the tolls minimizing TSTT compared to the least JAH statistic value obtained. This finding shows that tolls minimizing TSTT may also exhibit JAH behavior, though the extent of JAH for TSTT-minimizing profiles is always lower than the revenue-maximizing profiles.

4. Fourth, for the LBJ and MoPac networks with multiple access points to the ML, we observe that several toll profiles can cause violation of the speed limit constraint. However, the toll profiles optimizing the revenue or TSTT generate `%-violation` less than 2% for both MoPac and LBJ networks.

5. Last, the trends in throughput depend on the congestion level; if all vehicles clear at the end of simulation, throughput is a constant value equal to the number of vehicles using the system. However, for SESE and MoPac networks congestion persists till the

end of simulation. For the MoPac network, tolls generating high revenue have less throughput and the tolls generating low TSTT have a higher throughput.

Next, we seek toll profiles that optimize two objectives. Multi-objective reinforcement learning is an area that focuses on the problem of optimizing multiple objectives [82]. There are two broad approaches for solving this problem: single-policy approach and multi-policy approach. Single-policy approaches convert the multi-objective problem into a single objective by defining certain preferences among different objectives like defining a weighted combination of multiple objectives. Multi-policy approaches seek to find the policies on the Pareto frontier of multi objective. In this chapter, we focus on the single-policy approach due to its simplicity. We consider the weighted-sum and threshold-penalization approaches explained next.

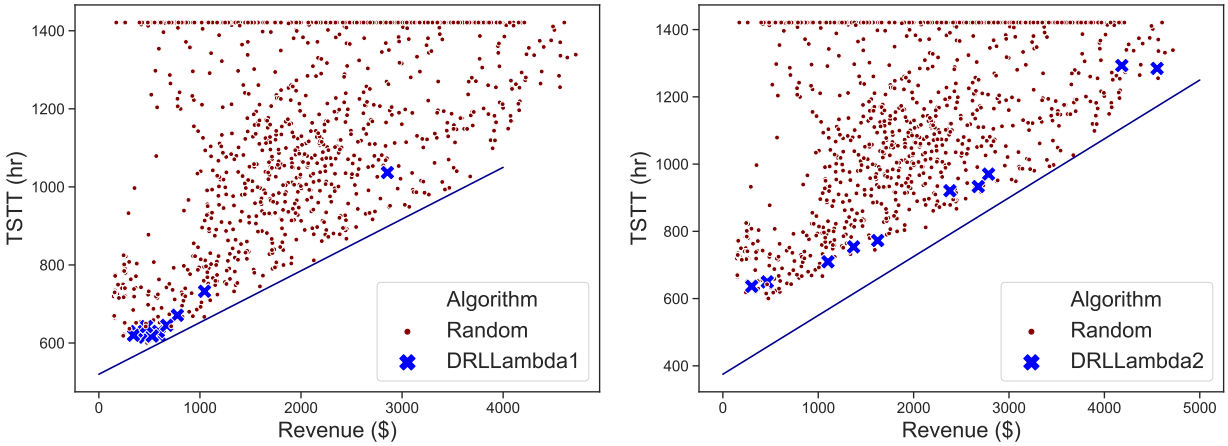
First, we apply the weighted-sum approach for finding a single policy that jointly optimizes TSTT and revenue. We define a new joint reward function $r^{\text{joint}}(s, a)$ as a linear combination of two rewards:

$$r^{\text{joint}}(s, a) = \lambda \; r^{\text{RevMax}}(s, a) + r^{\text{TSTTMin}}(s, a). \tag{4.21}$$

The value of λ is the relative weight of revenue (\$) with respect to TSTT (hrs) and has units hr/\$. Geometrically, λ represents the slope of a line on the TSTT-Revenue plot.

We run VPG and PPO algorithms for the new reward on the LBJ network with two different values of λ: $\lambda_1 = 0.1325$ hr/\$ and $\lambda_2 = 0.175$ hr/\$ (the values are chosen so that toll profiles in the mid-region of the TSTT-revenue plot are potentially optimal). Figure 4.12 shows the plot of optimal toll profiles obtained from Deep-RL algorithms on the TSTT-Revenue space. The slopes of the lines, equal to the λ values, are also shown, and the lines are positioned by moving them from the bottom to the top till they touch the first point among the generated space of points (that is, the line is approximately a tangent to the Pareto frontier).

As observed, Deep-RL algorithms are able to learn toll profiles that maximize the joint reward. For the λ_1 case, toll profiles are generated very close to the Pareto frontier; however, they are concentrated in the region where both TSTT and revenue are lower indicating the



(a) TSTT vs Revenue λ_1.

(b) TSTT vs Revenue λ_2.

Figure 4.12: Plot of TSTT vs revenue for the LBJ network for toll profiles generated randomly and toll profiles generated after optimizing the joint reward for two different values of λ

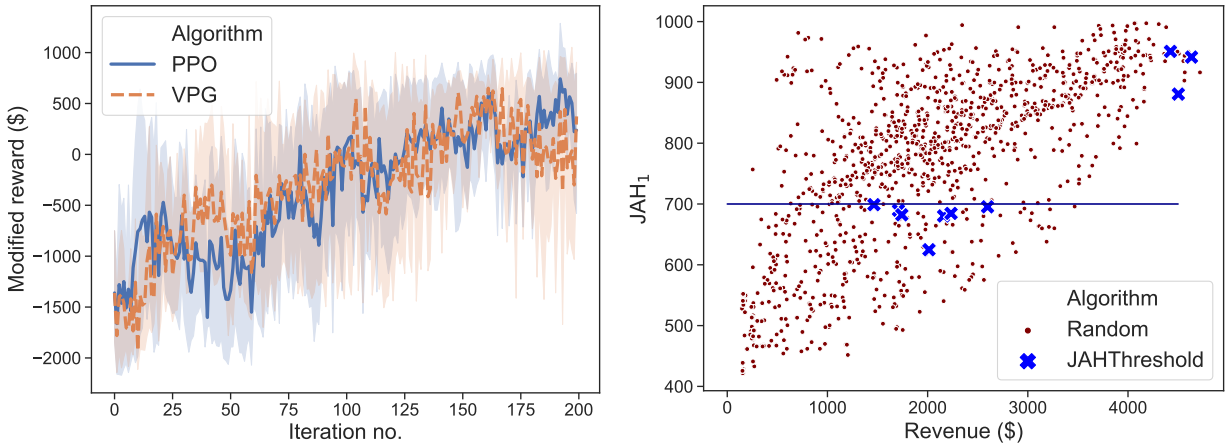
presence of local minima in the region. For the λ_2 case, the toll profiles are more spread out in terms of their values of TSTT and revenue; however, there are still a few toll profiles that are closer to the Pareto frontier tangent line which the Deep-RL algorithms did not find. This can again be explained by the behavior of policy gradient algorithms which are prone to converge to local optimum because they follow a gradient-descent approach.

Optimizing using a joint reward definition as Equation (4.21) can also be interpreted as following: that a toll operator is willing to sacrifice \$1 revenue for a $1/\lambda$ hours decrease in TSTT value. For the two values of λ, λ_1 and λ_2, this is equivalent to sacrificing \$1 revenue for a 7.55 hours and 5.72 hours decrease in total delay for the system, respectively. If they trade off these objective outside this range, the optimal policy will be the same as solely maximizing revenue or minimizing TSTT.

The second approach for solving multi-objective optimization problem is the threshold approach where we find toll policies that maximum revenue (minimize TSTT) such that TSTT (revenue) is less (higher) than a certain threshold. In this chapter, we apply the threshold-penalization method to model threshold constraints. This method simulates a policy and if at the end of an episode the constraint is violated, a high negative value is added to the reward to penalize such update. We test this technique to find tolls that

maximize revenue such that JAH_1 statistic is less than a threshold value. We use JAH_1 statistic because it has a physical interpretation and, unlike JAH_2, is not unitless.

We conduct tests for the threshold-penalization technique on the LBJ network with a threshold JAH_1 of 700 vehicles and add a reward value of $-\$3000$ to the final reward if at the end of simulation the JAH_1 statistic is higher than the threshold. Figure 4.13a shows the learning curve plotting the variation of modified reward with iterations. We observe that both VPG and PPO algorithms improve the modified reward with iterations, though it is hard to argue that they have converged. Learning is difficult in this case due to the same *credit assignment problem* where it is unclear will toll over an episode resulted in the constraint violation. Figure 4.13b shows the plot for tolls obtained from threshold-penalization technique on the JAH_1-Revenue space.



(a) (b)

Figure 4.13: (a) Plot of average modified reward with iteration while maximizing revenue with a reward penalty of $-\$3000$ if the JAH_1 statistic is more than 700 vehicles, and (b) the plot of JAH_1 vs revenue for the best-found toll profiles from the threshold-penalization method, along with toll profiles generated randomly

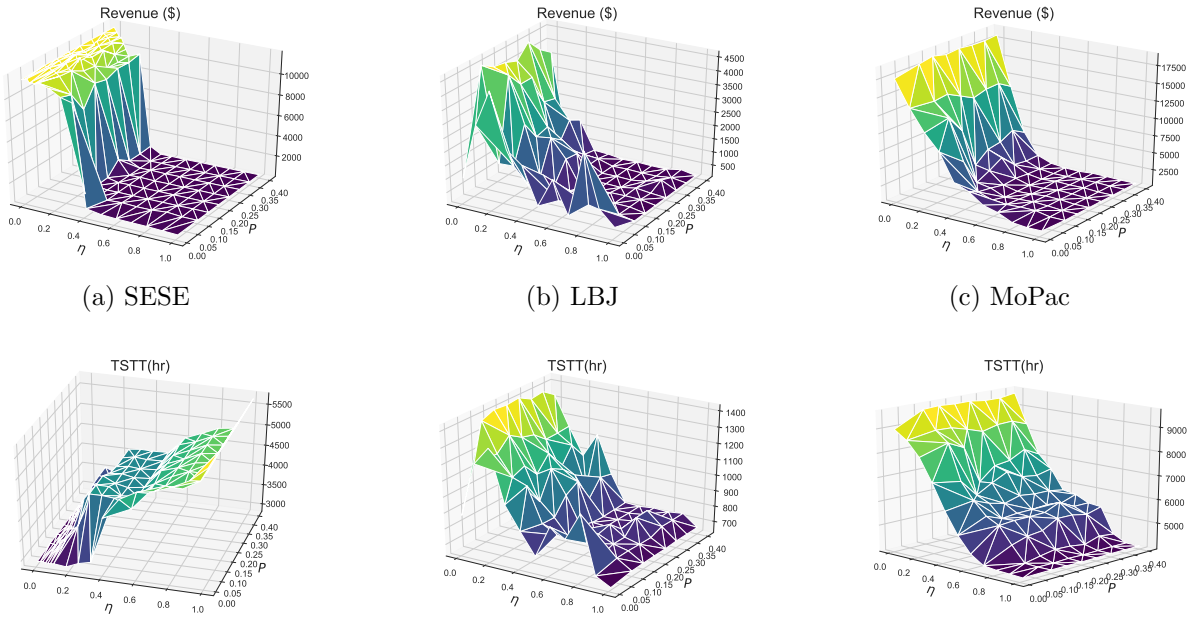
As observed, the threshold-penalization method is able to learn toll profiles with desired JAH value for 7 out of 10 random seeds. However, the learned toll profile is not the best found (that is, there are toll profiles with JAH less than 700 but generating revenue higher than \$2800, which is the best found revenue). This is because the modified reward did not converge (yet) after 200 iterations. Despite the lack of convergence, we conclude that the penalization method is a useful tool to model constraints on toll profiles. The success of

threshold-penalization method depends on the random seed, as that determines which local minimum the algorithm will converge to.

4.4.5 Comparison with other heuristics

In this section, we compare performance of the Deep-RL algorithms against other methods.

First, we study the variation of different objectives from the feedback control heuristic for different values of η and P values to identify the best performance for benchmarking. Figure 4.14 shows the variation of revenue and TSTT values for the SESE, LBJ, and MoPac networks. The values for each combination of parameters are reported as an average over 10 random seeds where the initial tolls on all toll links are set randomly between the minimum and maximum values for different seeds.



(a) SESE (b) LBJ (c) MoPac

(d) SESE (e) LBJ (f) MoPac

Figure 4.14: Variation of revenue ((a),(b),(c)) and TSTT ((d),(e),(f)) for different values of η and P parameters for the feedback control heuristic tested on SESE, LBJ, and MoPac networks

As observed, low values of η generate the highest average revenue across all combinations. Lower values of η ensure that ML is kept relatively more congestion free than the case

when η value is high. A low value of η charges high toll in the beginning and ensures that GPLs are more jammed promoting jam-and-harvest nature and generating more revenue.

In contrast to this, low values of TSTT are obtained for high values of η for both LBJ and MoPac networks. This is also intuitive: tolls minimizing TSTT operate the managed lane close to its critical density at all times. The contrary behavior of the SESE network, where low values of η also generate low TSTT, is because of the reasons explained in Section 4.4. For a given value of η, the variation of TSTT and revenue with P is not significant, indicating that the performance of feedback control heuristic is more sensitive to the η parameter.

Next, we compare the performance of feedback control heuristic against Deep-RL algorithms. Table 4.5 shows the values of different statistics reported as five-tuple: (revenue, TSTT, JAH_1, JAH_2, `%-violation`) for both the revenue maximization and the TSTT minimization objectives for Deep-RL algorithms (we report the better objective value between VPG and PPO) and the feedback control heuristic. We highlight the value of the optimization objective in bold. We also include the standard deviation in the objective value for both algorithms; the Deep-RL algorithm generates stochastic objective values due to the stochastic nature of the policy, while the feedback control heuristic generates stochastic objective values for different random initializations, given values of η and P.

The Deep-RL algorithms always find tolls with better objective values compared to the feedback control heuristic. For the revenue maximization objective, the average revenues from Deep-RL are 0.07–9.5% higher than the ones obtained from the feedback control heuristic. Similarly, for the TSTT minimization objective, the average TSTT values obtained from the Deep-RL algorithm are 0.09–10.38% lower than the average TSTT from the feedback control heuristic. Similar to the observations made earlier, the tolls maximizing the revenue also generate a high value of JAH_2 statistic and the tolls generating high revenue generate low TSTT (with an exception of SESE network). The value of `%-violation` on the ML is less than 2% on an average for all toll profiles, with insignificant differences between the Deep-RL algorithm and the feedback control heuristic.

Last, we also compare the performance of Deep-RL against VFA and SparseV algorithms which assume full observability. These algorithms rely on look-up table representation

Table 4.5: Comparison of Deep-RL against the feedback control heuristic for the two optimization objectives. Results are reported as a five-tuple: (revenue, TSTT, JAH_1, JAH_2, %-violation)

Revenue maximization objective		
	Deep-RL	**Feedback Control**
SESE	(**$11889.80 ± 3.77**, 2933.88 hr, 1166.43 veh, 0.34, 0%)	(**$11881.70 ± 7.92**, 2933.70 hr, 1166.44 veh, 0.34, 0%)
DESE	(**$497.97 ± 4.94**, 221.52 hr, 159.47 veh, 0.32, 0%)	(**$489.08 ± 0**, 223.26 hr, 160.43 veh, 0.32, 0%)
LBJ	(**$4718.43 ± 255.70**, 1396.15 hr, 986.81 veh, 0.49, 1.62%))	(**$4307.74 ± 275.59**, 1356.89 hr, 929.57 veh, 0.43, 0.77%)
MoPac	(**$18740.40 ± 61.64**, 9618.04 hr, 3102.17 veh, 0.32, 1.26%)	(**$18544.77 ± 133.36**, 9600.08 hr, 3097.71 veh, 0.32, 1.28%)
TSTT minimization objective		
	Deep-RL	**Feedback Control**
SESE	($11705.9, **2894.27 ± 16.22 hr**, 1166.38 veh, 0.34, 0%)	($11530.38, **2897.41 ± 18.72 hr**, 1166.53 veh, 0.34, 0%)
DESE	($271.46, **191.40 ± 7.53 hr**, 128.23 veh, 0.22, 0%)	($275.91, **213.57 ± 5.64 hr**, 128.00 veh, 0.25, 0%)
LBJ	($254.43, **641.72 ± 15.67 hr**, 541.18 veh, 0.25, 0.24%)	($158.46, **661.40 ± 0 hr**, 421.67 veh, 0.21, 0.32%)
MoPac	($655.45, **4022.45 ± 4.21 hr**, 1199.22 veh, 0.11, 0.07%)	($606.01, **4024.83 ± 11.01 hr**, 1141.37 veh, 0.11, 0.03%)

of value functions and discretize the state space[4]. For uniform comparison with the previous studies which use distance-based tolling [21, 63], we conduct tests only on DESE and LBJ networks with toll values varying between $0.05/mile and $0.8/mile. Additionally, since the SparseV algorithm in the previous chapter is only defined for the revenue-maximization objective, we only focus on that objective. Table 4.6 shows the best found revenue for the two networks using the three algorithms. As observed, Deep-RL algorithm outperforms VFA and SparseV by generating an average 11.85% percent higher revenue. These findings show that even under partial observability Deep-RL algorithms can learn toll profiles with better objectives than algorithms assuming full observability.

[4]It is possible to implement neural networks as function approximators in VFA and SparseV than using look-up tables; however, our focus is on direct comparison with previous algorithms in the literature.

Table 4.6: Comparison of best-found toll objective from Deep-RL algorithm with partial observability against the VFA and SparseV heuristics that assume full observability

Revenue-maximization best-found revenue ($)			
	Deep-RL	**VFA**	**SparseV**
DESE	503.71	500.18	493.80
LBJ	4634.69	2880.59	4316.03

4.5 Summary

In this chapter, we developed Deep-RL algorithms for dynamic pricing of MLs with multiple access points. We showed that the Deep-RL algorithms are able to learn toll profiles for multiple objectives, even capable of generating toll profiles lying on the Pareto frontier. The average objective value converged within 200 iterations for the four networks tests. The number of sensors and sensor locations were found to have little impact on the learning due to the spatial correlation of congestion pattern. We also conducted transferability tests and showed that policies trained using Deep-RL algorithm can be transferred to setting with new demand distribution and VOT distribution without losing performance; however, if the lane choice model is changed the transferred policy performs poorly. We analyzed the variation of multiple objectives together and found that TSTT-minimizing profiles may be similar to revenue-maximizing profiles for certain network characteristics where the GPL invariably becomes congested early in the simulation. We also compared the performance of Deep-RL algorithms against the feedback control heuristic and found that it outperformed the heuristic for the revenue maximization objective generating average revenue up to 9.5% higher than the heuristic and generating average TSTT up to 10.4% lower than the heuristic.

The Deep-RL model in this chapter requires training, which is dependent on the input data and the parameters. We make following implementation recommendations. If a toll operator has access to the input data including the demand distribution and driver lane choice behavior, we recommend first calibrating a lane-choice model using the data and then using the calibrated model to train the policy for the desired objective under desired constraints. If the driver lane choice data is very detailed and can exactly identify how many travelers chose the ML at each time, then that data can be directly used in training without calibrating a lane-choice model; however, a calibrated model is still recommended as it can

assist in conducting sensitivity analysis to other inputs and/or long-term planning. If the input data is not available or has poor accuracy, we recommend two alternatives. A toll operator can either train the Deep-RL model considering high stochasticity by choosing a large values for the standard deviations (σ_d and σ_o), or train several policies for different combinations of inputs and use the policy based on the expected realization of inputs from field data for real-time implementation. Lastly, we also recommend retraining the toll policy using real-time data. For example, a policy can be trained from the historic data and then improved based on the observations from a specific day and the improved policy can then be applied to the next day. Additionally, though the model in this chapter trains a stochastic policy, for implementation purposes, we can use a deterministic policy with the tolls set as the mean value predicted by the policy.

Chapter 5

Static Multiclass User Equilibrium with Recourse

Given that travelers can change their routes online, we assume travelers to follow a fixed policy from their origin to their destination, instead of a fixed path. A policy determines what next node to choose given the current received information so far. The user-equilibrium with recourse concept proposed in Unnikrishnan and Waller [15] seeks to find equilibrium flows using these policies such that all used policies between an origin and a destination have equal and minimal expected costs. For managed lane networks, travelers have varying values of time and perceive different costs for the same policy, where the perceived cost is a linear combination of the expected travel time and expected toll. In such cases, we need to construct revised equilibrium models where travelers from multiple classes participate in the congestion.

The current literature on multi-class assignment is rich in algorithms for solving multiclass multicriteria traffic assignment for large scale networks [83, 84, 85, 86]. Dial [83, 84] developed efficient algorithms for solving multi-criteria traffic assignment for continuous distribution of the value of time (VOT). The user equilibrium condition in these settings state that each traveler between the same origin and destination having the same value of time use paths which have equal and minimal costs.

An extension of the same to the case where travelers choose policies instead of fixed routes can be stated as the principle of multiclass user equilibrium with recourse: at user equilibrium all used policies between an origin and a destination used by the vehicles of the same VOT class have equal and minimal costs. Figure 5.1 shows the schematic for the M-UER model and its relation to other models in the literature.

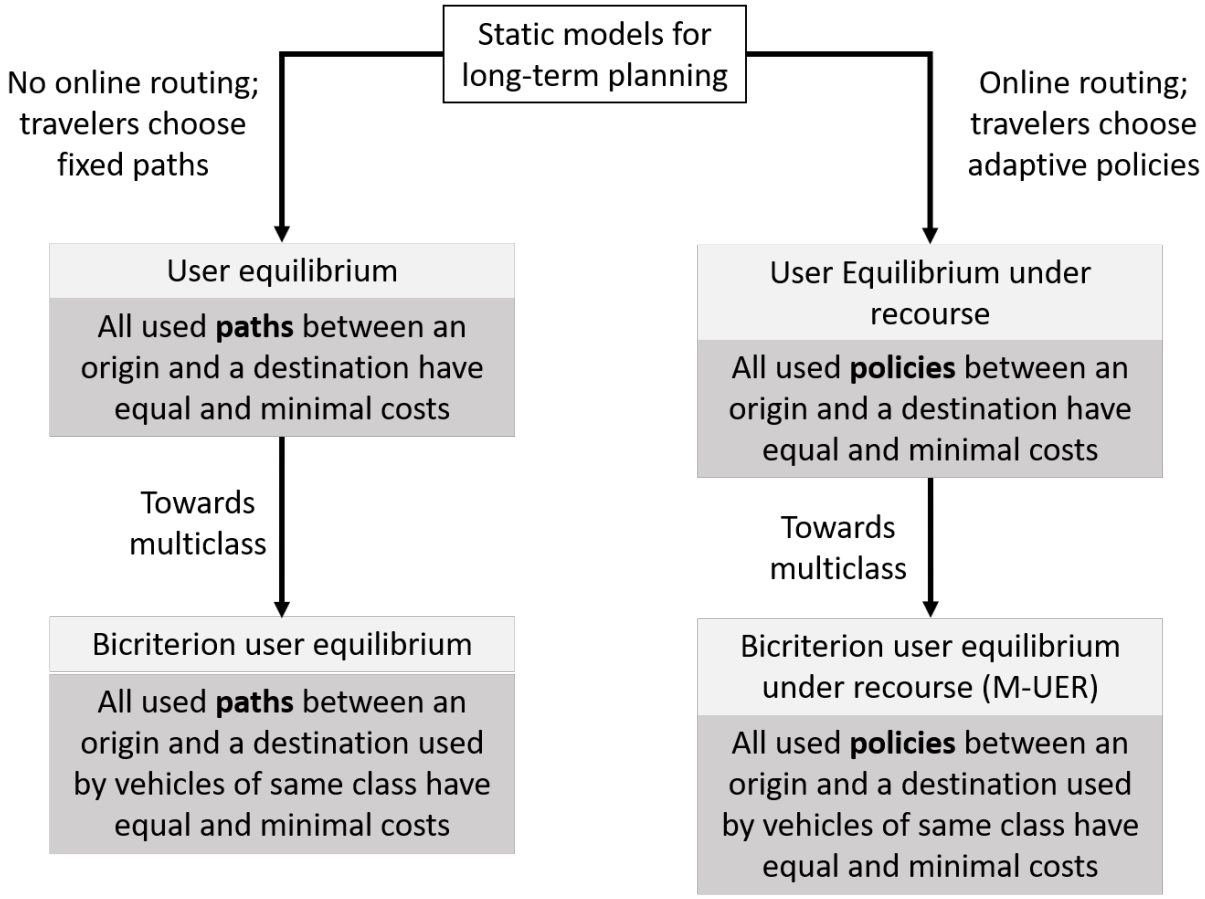


Figure 5.1: Schematic of M-UER model and its relation to other models in the literature

5.1 Literature review

User equilibrium formulation for several travelers under the influence of congestion is commonly employed to study long-term driver behavior. Yang ang Huang [85] classify the multiclass models in the traffic assignment literature into two categories:

1. Travelers of each class perceive the link travel time functions to be different but the costs are based on the total flow of the link [87]. A common example for this includes models with car and truck vehicles where the link travel time function for the different types of vehicles is different.

2. Travelers of each class have same link travel time or toll function, but differ in their calculation of perceived costs due to a different value of a parameter. The commonly used parameter is the value of time. The research in this area has focused on discrete [86, 88] and continuous VOT distributions [83, 84], separately.

The primary focus of this chapter is to extending these models for equilibrium with recourse. The current state-of-the-art for UER models includes algorithms from acyclic networks in [15] and cyclic networks in Rambha et al. [16]. The idea of policy based routing has also been extended to the transit assignment [89]. In this chapter, we propose a variational inequality for M-UER and use algorithms from the traffic assignment literature to solve M-UER policy flows for discrete and continuous VOT distributions.

5.2 M-UER model

5.2.1 Assumptions

For our model, we make following assumptions:

1. Each link exists in a finite discrete number of states with a fixed probability of occurrence for each state.

2. The total cost on each link is a linear combination of travel time and toll on the link, and does not depend on any factor outside of travel time and toll.

3. Each traveler behaves rationally and is fully aware of the probability distribution of travel time and toll across the network. Though a limiting assumption for technologies currently in place, we can expect travelers to be more aware in the future with automated data recording by the connected and autonomous vehicles.

4. Link states are independent of each other. This might be a restrictive assumption in real-life setting where link travel times are spatially and temporally correlated. However, we sacrifice realism in our model by trading off with models which are theoretically sound and tractable. Exploring link travel time and toll correlation is left for the future work.

5.2.2 Model

In this section we introduce the notation for the M-UER model. We borrow the notations used in explaining the UER model in Rambha et al. [16].

Let $G = (N, A)$ denote a strongly connected network with N as the set of nodes and A as the set of arcs. Let $\Gamma(i)$ and $\Gamma^{-1}(i)$ denote the sets of upstream and downstream nodes of node i respectively. Let W denote the set of all origin-destination pairs in the network with demand between them. Let K denote the set of all driver classes and a traveler of class $k \in K$ has a value of time α_k, where K is discrete and finite. We shall later relax this assumption by extending K to be an infinite set. For any $(u, v) \in W$, let d^k_{uv} denote the demand of class k from origin u to destination v.

Each arc $(i, j) \in A$ is assumed to exist in one of the states in set S_{ij} where probability that link (i, j) exists in state $s \in S_{ij}$ is denoted by p^s_{ij}. Let $x^{s,k}_{ij}$ denote the number of vehicles of class k using link (i, j) in state s. Let $t^s_{ij}(x^s_{ij})$ denote the travel time on link (i, j) in state s which is a function of total link flow x^s_{ij} in state s, where $x^s_{ij} = \sum_{k\in K} x^{s,k}_{ij}$. Similarly, define τ^s_{ij} to denote the toll charged on link (i, j) in state s. We assume tolls are constant and do not depend on link flows. We assume $t(\cdot)$ as an increasing and convex function of flow in any state (which is a reasonable assumption in the traffic assignment literature [90]). Let $S = \bigcup_{(i,j)\in A} S_{ij}$ denote the set of all link-states in the network. Then, we define link-state flow vector $\boldsymbol{x}$ with dimensions $|S| \times 1$ containing total link-state flow for each state, where we use the notation $|\cdot|$ for any set to denote the number of elements in the set.

A traveler receives online information upon arrival at a node and learns the travel time and toll information on the downstream links. Let $\theta \in \Theta_i = \times_{(i,j)\in A} S_{ij}$ denote the information vector received at node i, where Θ_i is the set of all possible information that can be received at node i. Let θ_{ij} denote the link-state of link (i,j) under information θ at node i. Let q_i^θ denote the probability of receiving message $\theta \in \Theta_i$, then assuming that the link travel time and toll updates on a link are independent of other links in the network, we get $q_i^\theta = \Pi_{(i,j)\in A} p_{ij}^{\theta_{ij}}$. Let $\phi = \{(i,\theta) \ : \ i \in N, \theta \in \Theta_i\}$ denote the set of node-states, which stores all possible node-states associated with node i.

We then formulate the problem as a finite horizon Markov decision process with no discounting and with a terminal state. The components of MDP are:

- **State:** Define the set of node-states $\phi = \{(i,\theta) \ : \ i \in N, \theta \in \Theta_i\}$ as the state of the MDP
- **Action:** The available action in each time node state (i,θ) is given by the set of downstream nodes to the current node, $\Gamma(i)$
- **Transition probability:** Given node state (i,θ) and action (i,j), the probability of transitioning to another node state $(j,\bar{\theta})$ is given by $q_j^{\bar{\theta}}$ and is zero for all other node states.
- **One step cost:** The one step cost of taking an action is the linear combination of toll and travel time given a vehicle's VOT. Note that the one step cost varies with different classes.

Define a policy $\pi : \phi \to N$ as a function that maps each node-state to a downstream node or a terminal node if the node is a destination. Since the network is acyclic, the policy always terminates at the destination node. Let Π_v denote the set of policies terminating at node v and $\Pi = \bigcup_{v\in Z} \Pi_v$ be the set of all policies in the network across different destinations.

We explain these notations using an example. Consider the network shown in Figure 5.2, with two routes connecting nodes 1 and 3. Link $(1,3)$ exists in two states with given link performance function. Link $(1,2)$ also exists in two states with fixed travel time but different toll, and link $(2,3)$ exists in only one state.

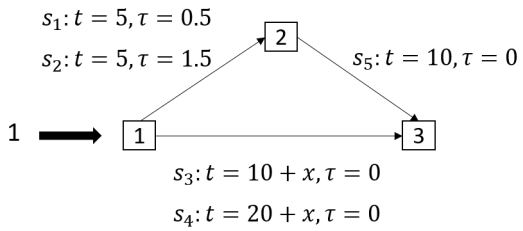


Figure 5.2: Sample network

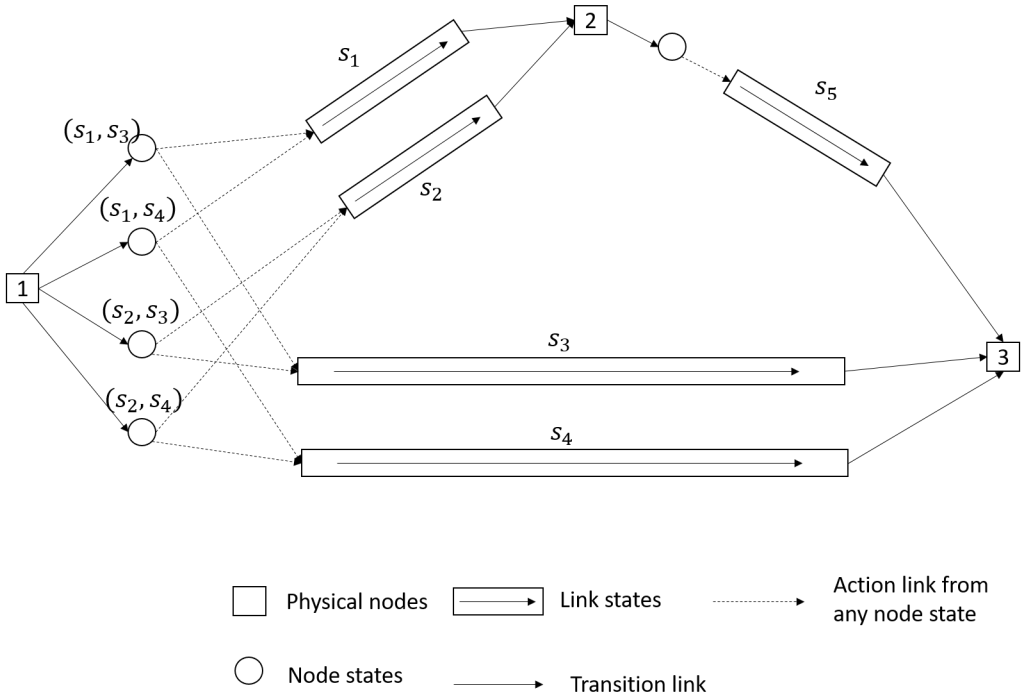


Figure 5.3: Network transformation for the network in Figure 5.2 by splitting it into node-states, link-states, and physical nodes

We present a network transformation, similar to the one in Boyles [91] as shown in Figure 5.3. Each physical node connects to a node-state which in turn connect to link state through action links which connect to the downstream physical node.

We define $\rho_{ij}^{s\pi}$ as the probability of leaving node i via link (i, j) in state $s \in S_{ij}$, which can be evaluated using Equation (5.1).

$$\rho_{ij}^{s\pi} = \sum_{\theta \in \Theta_i \ : \ \pi(i,\theta)=j, \ \theta_{ij}=s} q_i^{\theta} \tag{5.1}$$

Each traveler follows a policy from their origin to their destination and seeks to

minimize the total cost for the chosen policy. The travel times t_{ij}^s and tolls τ_{ij}^s are assumed fixed for routing of a single traveler. We define following cost-to-go values for each location point in the transformation map:

- $G_i^{\pi,k}$ as the expected travel cost from node i to destination v for a traveler of VOT α_k following policy π or the cost-to-go from a physical node
- $G_{ij}^{s,\pi,k}$ as the expected travel cost starting the upstream end of link (i,j) in state s for a traveler with VOT α_k following policy π
- $G_{(i,\theta)}^{\pi,k}$ as the expected travel cost starting the node state (i,θ) in state s for a traveler with VOT α_k following policy π.

For deterministic policies (which maps deterministically to a downstream link), $G_{(i,\theta)}^{\pi,k} = G_{\pi(i,\theta)}^{\theta_{ij},\pi,k}$. Since the action and state space is finite and one step rewards bounded, there is atleast one optimal deterministic policy. We focus our attention on deterministic policies. In the terms of the MDP literature, $G_i^{\pi,k}$ is the expected cost-to-go from node i for a policy $\pi \in \Pi$. These cost-to-go values under policy π can be evaluated using a recursive structure as shown below:

$$G_v^{\pi,k} = 0 \tag{5.2}$$

$$G_i^{\pi,k} = \sum_{j \in \Gamma(i)} \sum_{s \in S_{ij}} \rho_{ij}^{s\pi} (\alpha_k t_{ij}^s + \tau_{ij}^s + G_j^{\pi,k}) \quad \forall i \in N/\{v\} \tag{5.3}$$

This recursive relation can be used to solve the proposed MDP to find optimal node-state values for each node state and the optimal policy for each class k using the Bellman equation.

For the M-UER formulation, however, we work with the cost-to-go values for each link state which can be expressed as following function of cost-to-go for each node state:

$$G_{ij}^{s,\pi,k} = \alpha_k t_{ij}^s + \tau_{ij}^s + G_j^{\pi,k} \qquad \forall (i,j) \in A, s \in S_{ij}, k \in K \tag{5.4}$$

which, upon eliminating the cost-to-go for node states gives us the following relation:

$$G_{ij}^{s,\pi,k} = \alpha_k t_{ij}^s + \tau_{ij}^s + \sum_{h \in \Gamma(j)} \sum_{\bar{s} \in S_{jh}} \rho_{jh}^{\bar{s}\pi} G_{jh}^{\bar{s},\pi,k} \tag{5.5}$$

This equation can be written in an equivalent matrix form for each VOT class $k \in K$:

$$\mathbf{G}^{\pi,k} = \alpha_k \mathbf{t} + \boldsymbol{\tau} + \mathbf{P}_\pi \mathbf{G}^{\pi,k} \tag{5.6}$$

where $\mathbf{G}^{\pi,k}$, $\mathbf{t}$, and $\boldsymbol{\tau}$ are vectors of dimensions $|S| \times 1$ and $\mathbf{P}_\pi$ is the probability matrix of dimensions $|S| \times |S|$ containing elements $\rho_{ij}^{s\pi}$.

Let $y_i^{\pi,k}$ denote the number of travelers of VOT class k who originate from node i and follow policy π. Flow conservation requires that $y_i^{\pi,k} \geq 0$ for all origin nodes i, policies π and VOT class k, and the sum of total flow across all policies is equal to the demand, that is $d_{uv}^k = \sum_{\pi \in \Pi_v} y_i^{\pi,k}$, where a destination based aggregation of policies has been employed.

Next, we relate the policy flow to flow on each link-state using Equation (5.7),

$$x_{ij}^{s,\pi,k} = \rho_{ij}^{s\pi} y_i^{\pi,k} + \rho_{ij}^{s\pi} \sum_{h \in \Gamma^{-1}(i)} \sum_{\bar{s} \in S_{hi}} x_{hi}^{\bar{s},\pi,k} \tag{5.7}$$

where the first term is the total flow originating at the from node that will choose the link state s, while the second term is the total flow arriving at the tail node from its predecessor link states.

The Equation (5.7) can be represented in matrix form as:

$$\mathbf{x}^{\pi,k} = \mathbf{b}^{\pi,k} + \mathbf{P}_\pi^T \mathbf{x}^{\pi,k}$$

where $\mathbf{x}^{\pi,k}$ is a $|S| \times 1$ vector of link-state flow wrt each VOT class, $\mathbf{b}^{\pi,k}$ is a $|S| \times 1$ vector of flow departing from physical node i following policy π for each VOT class with elements $\rho_{ij}^{s\pi} y_i^{\pi,k}$.

The M-UER principle says that all feasible policy flows $\mathbf{y}$ at equilibrium satisfy the

following property:

$$y_u^{\pi,k} > 0 \;\Rightarrow\; G_u^{\pi,k}(\boldsymbol{y}) = \min_{\pi' \in \Pi_v} G_u^{\pi',k}(\boldsymbol{y}) \quad \forall v \in Z, \pi \in \Pi, k \in K \tag{5.8}$$

Similar to the earlier formulations, this intuitive equilibrium property can be converted to an optimization problem as following. The objective function is a linear combination of convex functions and is thus convex.

$$\min_{\mathbf{y},\mathbf{x},\mathbf{x}^{\pi,k},\mathbf{b}^{\pi,k}} \sum_{(i,j)\in A} \sum_{s\in S_{ij}} \int_0^{\sum_{k\in K} x_{ij}^{s,k}} \alpha_k t_{ij}^s(\boldsymbol{x})\, dx + \sum_{(i,j)\in A} \sum_{s\in S_{ij}} \sum_{k\in K} x_{ij}^{s,k} \tau_{ij}^s \qquad \textbf{(M-UER)} \tag{5.9}$$

$$\text{s.t.} \sum_{\pi\in\Pi_v} y_u^{\pi,k} = d_{uv}^k \qquad \forall (u,v) \in W, k \in K \tag{5.10}$$

$$\sum_{\pi\in\Pi} x_{ij}^{s,\pi,k} = x_{ij}^{s,k} \qquad \forall (i,j) \in A, s \in S_{ij}, k \in K \tag{5.11}$$

$$(I - \mathbf{P}_\pi)\mathbf{x}^{\pi,k} = \mathbf{b}^{\pi,k} \qquad \forall \pi \in \Pi \tag{5.12}$$

$$y_u^{\pi,k} \geq 0 \qquad \forall \pi \in \Pi, u \in W, k \in K \tag{5.13}$$

Proposition 3. *The optimal solution of the convex program in* (5.9)-(5.13) *corresponds exactly to the policy flows satisfying the MUER definition in Equation* (5.8)

Proof. The proofs follows logically from Rambha et al. [16] by simply discretizing policy flows and link-state flows by VOT class. □

Additionally, since the cost functions are positive and strictly increasing in space of total link-state flow, the formulation results in unique total link-state flows.

5.3 Solution algorithms

The online shortest path MDP is solved using backward recursion by starting at the terminal node and moving in the reverse topological order. The structure is similar to the algorithm in Chapter 2.

To solve the M-UER problem, we use the method of successive averages (MSA), which is a link-state-based algorithm. This method starts with loading all travelers on the shortest policy. And, then it iteratively combines the link-state flows, where in each iteration the

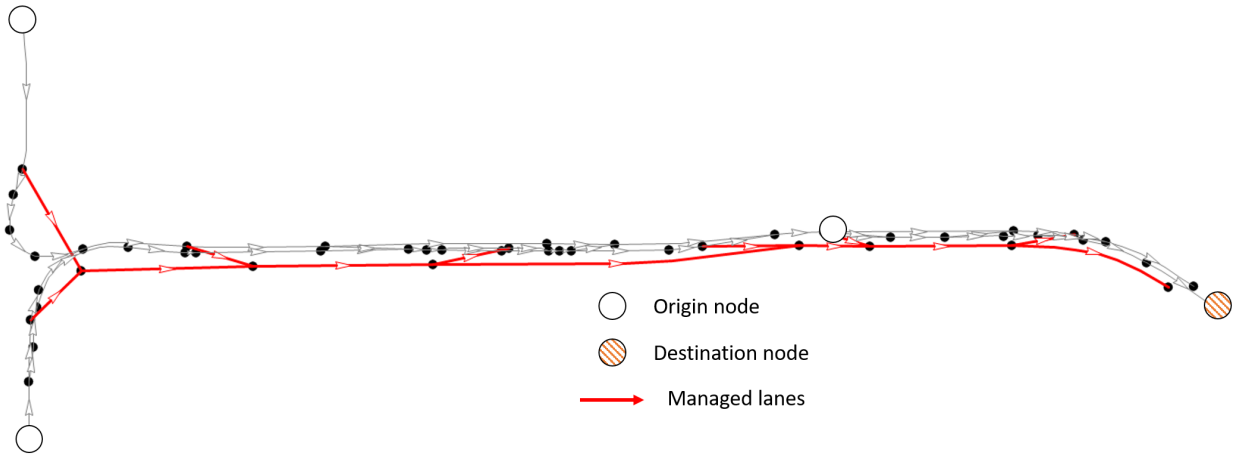


Figure 5.4: Network for North Tarrant Expressway for eastbound direction of toll segment 1

new set of shortest hyperpath flows are given a weightage of $1/n$, where n is the iteration number. This algorithm operates in the space of link-states and is thus tractable and has less memory requirements.

5.4 Results

We test the proposed algorithms on the North Tarrant Expressway network in Dallas, TX with 54 nodes and 67 links. Figure 5.4 shows the network. The network was extracted from the original demand model provided by the North Central Texas Council of Government. The freeflow travel time and capacity of each link were used as is and the travel time function was assumed to follow the standard non-linear BPR function. The tolls on each managed lane arc were as obtained from the field. The supply-side uncertainty in the network was added by creating two additional states on the links on general purpose lanes with revised free-flow travel times as 0.7 and 2 times the true value, and two additional states on the express lanes with tolls multiplied by 0.7 and 2 times the true value.

We first compare the rate of convergence for the MSA algorithm for varying levels of demand. Figure 5.5 shows the variation in the relative gap values with iteration. As observed, the value decrease with iteration number. The rate of decrease is faster for lower demand levels due to lower congestion in the network.

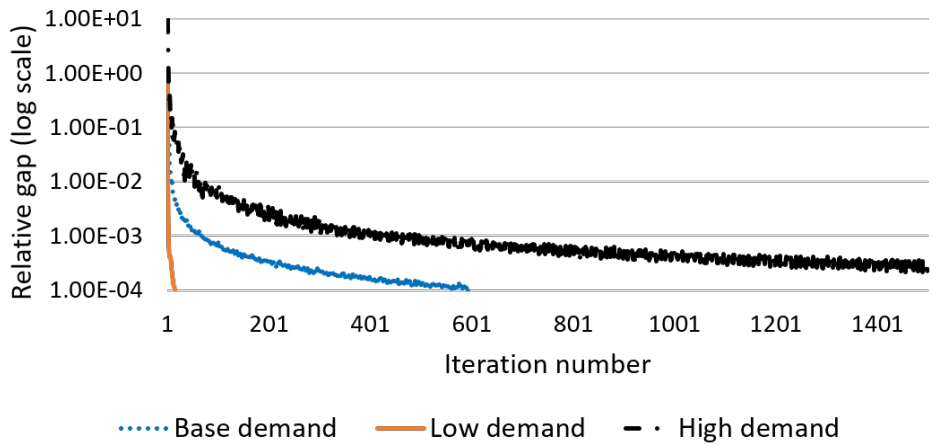


Figure 5.5: Convergence of relative gap for varying demand levels for the NTE network

We also compare the volume to capacity ratios obtained by M-UER and standard multiclass traffic assignment. For running standard multiclass assignment, we assigned the capacity, free flow travel time, and tolls on a link to be the weighted average of the respective values in different states. Figure 5.6 compares the v/c ratio for the two assignment procedures. The link flow for the M-UER runs is the weighted average of the total flow in each link state. As observed, M-UER procedure splits the flow among managed and general purpose lanes and generates less congestion, whereas the multiclass assignment sends more travelers to the general purpose lane and is not able to adapt their routes based on the received downstream information.

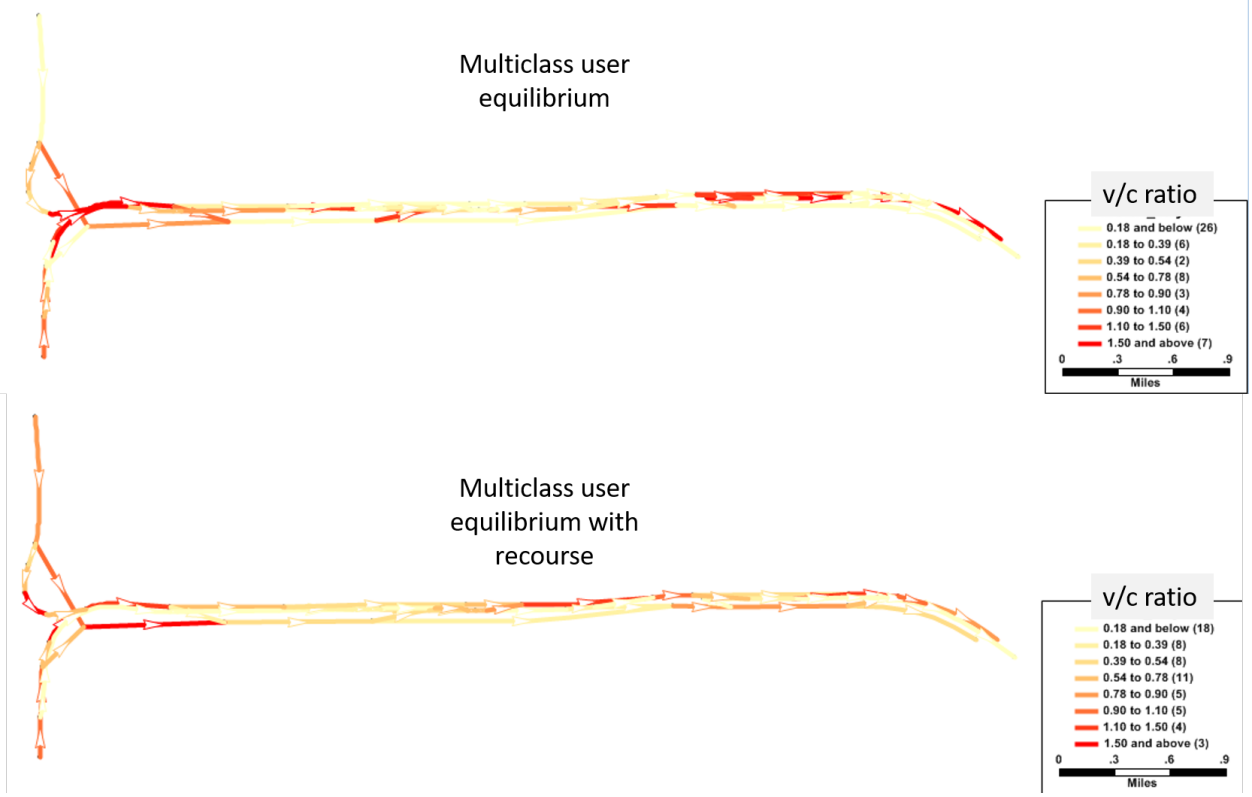


Figure 5.6: Comparison of v/c ratios between M-UER runs and runs based on multiclass traffic assignment for the NTE network

The total system expected travel cost obtained from solving M-UER is $93419.397 while the same from standard multiclass traffic assignment is $94924.043, which is 1.6% lower that the M-UER value. This shows the benefit obtained to the system if we incorporate online route choice behavior of the travelers. Additionally, the flow prediction on express lanes is 45% lower using standard multiclass traffic assignment. This indicates that ignoring adaptive route choice can have a significant impact on the long-term traffic forecast, useful for evaluation of future express lane facilities.

5.5 Summary

In this chapter, we have proposed a model for multiclass user equilibrium under recourse where travelers made adaptive changes to their route choice at each diverge location. M-UER is then reformulated as a convex program. Method of successive averages is used to solve the M-UER for test networks. Test on the North Tarrant Expressway network in Dallas,

TX show convergence of the relative gap measure with iterations. The M-UER link flows were found to be different from the flows obtained from static multiclass user equilibrium with total system travel cost differing by 1.6%. The proposed model is useful for predicting long term traffic predictions given that travelers respond to online information.

Chapter 6

Sensitivity Analysis of User Equilibrium with Recourse for Network Contraction

6.1 Introduction

Models for MLs can be divided into two broad categories: corridor-level models and network-level models. Corridor-level models are used for analysis made for the corridor like dynamic pricing, traffic operations, estimation of bottlenecks, etc. These models capture vehicle-to-vehicle interaction and provide detailed statistics like the number of vehicles per lane, distribution of speeds, and the location of bottlenecks [42]. On the other hand, network-level models are used for analysis on large-scale networks such as long-term traffic and revenue forecasts. Network-level models aggregate travel behavior and find steady-state traffic condition in a system with multiple travelers.

Despite their advantages, both these models have limitations. In particular, corridor-level models consider the travel demand to be inelastic and ignore the diversion of travelers away from or towards the ML after changes in corridor operations. Similarly, network-level models, in their traditional four-step planning format, assume tolls as static values and ignore the *en route* changes to the route choice of travelers made possible by the availability of real-time toll and travel time information.

In this chapter, we have two motivating questions: (a) how to simplify the integration of dynamic tolls and *en route* changes in static long-term planning models, and (b) how to approximate the diversion of demand (using elastic demand functions) for corridor-level models. The answers to these questions are critical to improve the accuracy of current models for MLs used for planning and operations of future ML installations.

We propose a solution to both these questions by determining sensitivity of steady-state traffic pattern on a managed lane corridor under the principle of user equilibrium with recourse (UER) that finds equilibrium over route choices of travelers under the presence of

uncertainty in tolls and travel times.

Sensitivity analysis of UER models allows network contraction of the ML corridor using artificial links whose parameters are estimated. For example, consider the managed lane corridor in Figure 6.1 where the thicker links are MLs and the highlighted nodes are locations where travelers make an *en route* lane choice decision. Using sensitivity analysis, we can condense this network into an artificial link connecting the origin with the destination by estimating derivative of the expected costs between origin and destination with respect to the variation in demand (this process is commonly referred as network contraction).

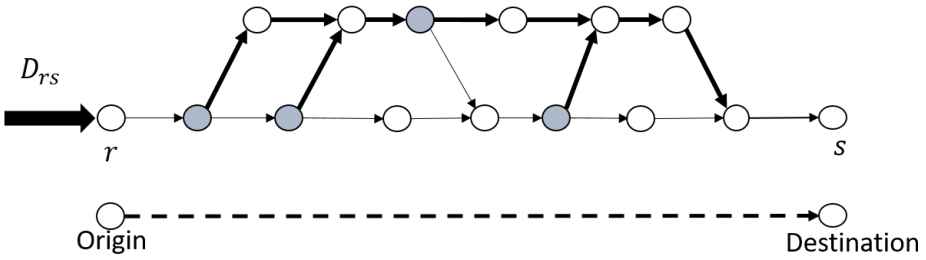


Figure 6.1: Approximating a ML corridor using an artificial link connecting the origin and the destination

While we have efficient algorithms to solve user equilibrium on large-scale networks [92, 93], the UER models lack scalability to large networks due to the possibility of cyclic route choice patterns at equilibrium [94]. However, the localized nature of supply-side uncertainty on express lanes allows us to replace the ML corridor using artificial links generated from network contraction, thus obviating the need for solving UER on the entire network. This integration of UER models in traditional equilibrium models can capture network-level impacts of stochastic tolls and adaptive driver behavior, thus answering the first motivating question. Furthermore, conducting an additional sensitivity analysis on the entire network with MLs approximated as an artificial link, we can determine the variation of demand using the corridor as a function of cost parameters, thus answering the second motivating question.

The primary contributions of this chapter are as follows:

1. We present a gradient-projection algorithm for generating computationally-efficient solutions to UER with better accuracy than the algorithms in the literature.

2. We present a convex program for sensitivity analysis of UER models and present an

extension of the gradient-projection algorithm above for computing the sensitivity parameters.

Though the sensitivity analysis model in this chapter is developed for ML corridors, the models can be easily extended for the sensitivity of UER models on any network with localized supply-side uncertainty such as non-recurring congestion due to incidents. The rest of this chapter is organized as follows. Section 6.2 reviews the literature and places our work in context of existing sensitivity analysis models. In Section 6.3, we describe the notations, assumptions, and the details of UER model for ML networks. Section 6.4 formulates the UER-sensitivity problem as a convex program and presents a solution algorithm based on the gradient projection algorithm. Section 6.5 presents experimental results on real-world networks. Last, Section 6.6 summarizes the chapter.

6.2 Related work

In this section, we first review the models for traffic assignment under the presence of uncertainty and motivate the need for UER models. Then, we review the literature on sensitivity analysis for traditional user equilibrium models and the commonly-used network contraction techniques.

6.2.1 Traffic equilibrium under stochasticity

Managed lanes are characterized by supply-side uncertainty where tolls are uncertain inducing stochasticity in lane choices among travelers. Several approaches have been used in the literature to address stochasticity in traffic assignment models. Classical stochastic user equilibrium (SUE) models assume that network parameters are deterministic, and the source of stochasticity is due to different perception in the generalized cost of routes by travelers [90, 95]. If the network parameters are random variables, the model replaces those parameters with their expected value. However, as argued in Wallace [96], replacing the stochasticity in network parameters with their deterministic equivalents (like the expected value) can result in suboptimal solutions.

An alternate approach to model stochasticity considers day-to-day evolution of route choice of travelers and analyzes the steady-state of route choices [97]. Day-to-day traffic

assignment models focus on the stability of equilibrium modeled as Markov chains. However, these models are computationally inefficient for large-scale networks because the state space is a function of the number of travelers and the action space is the total number of routes, and both those quantities can be arbitrarily large for a large-scale network. Approximation methods based on Monte-Carlo simulation have been proposed to approximate the steady-state flow distribution [98, 99] and it has been shown that for certain networks, the expected flow from day-to-day assignment models is equivalent to the SUE solution.

Both SUE and day-to-day assignment models assume that travelers follow a fixed path from their origin to their destination. However, this assumption for ML corridor translates to travelers choosing ML even if the tolls are really high or the GPL is uncongested. UER models overcome this issue by allowing travelers to adapt their routes *en route*. UER models assume that instead of following a fixed path, travelers follow a fixed policy from their origin to their destination, which determines what next node to choose given the current received information so far. The equilibrium for UER models is the flow distribution where all used policies between an origin and a destination have equal and minimal expected costs [15]. UER models have been used for applications including dynamic tolling under the presence of non-recurring congestion [16] and network design problems with uncertainty [100].

The models for UER are similar to the independently studied area of Markov decision process (MDP) routing games applied in the context of ridesharing where drivers choose which area to relocate given the real-time observation of demand and surge-prices [101]. MDP routing games, like UER models, are a special case of continuous population stochastic games where each infinitesimal agent solves an MDP with the rewards dependent on other agents' action[1]. The fundamental difference between MDP routing games and UER models is the way congestion is modeled. UER models consider congestion on links which represent physical connections between different parts of a network. On the contrary, MDP routing games consider congestion on abstract "links" connecting different states with congestion as a function of the number of agents taking a specific action in a state. Because UER models

[1]We note that solving equilibrium under these settings can be formulated as a multi-agent reinforcement learning (MARL) problem [54] with competition among infinitesimal agents; however, MARL techniques provide only an approximate solution. In contrast, MDP routing games derive analytical properties of the equilibrium which are useful for theoretical sensitivity analysis

capture uncertainty in physical transportation infrastructure, we consider these for modeling MLs.

6.2.2 Sensitivity analysis and network contraction

Sensitivity analysis of equilibrium models determines the variation in the equilibrium solution (that is, the flow on each link) with respect to changes in demand and network parameters. For traditional user equilibrium problems, variational inequality method has been proposed which uses the implicit-function theorem to determine the derivatives of link flows to perturbations in demand and link cost parameters [102]. Patriksson [103] derived conditions for existence of these derivatives and showed that the directional derivatives and gradients can be obtained by solving an optimization problem which is identical to the standard traffic assignment problem except with linear costs, called the linearization approach. Expanded further in Josefsson and Patriksson [104] and Jafari and Boyles [105], this linearization approach allows applying existing algorithms developed for solving user equilibrium towards solving sensitivity parameters and simplifies the network design problems where the sensitivity analysis is conducted in each iteration. Other methods for sensitivity analysis have focused on multiple driver classes [106], though the computational performance is poor for a higher number of classes. In this chapter, we focus on single driver class, that is, all drivers have identical willingness to pay.

The most relevant work in the recent literature is by Li et al. [107] where the authors conduct sensitivity of MDP routing games with respect to changes in state-action costs, which are equivalent to the sensitivity to link performance function parameters in each link-state in our framework. Methods based on inverting the transition probability matrices are used to derive the sensitivity parameters[2]. However, the inversion of matrices which have dimensions as large as the number of node states can be computationally challenging. The sensitivity analysis is applied on a five-node stochastic Braess-type network. In this chapter, we exploit the property that the routing MDP for each traveler is associated with an acyclic network and derive a convex program for determining sensitivity parameters, which can be

[2]These matrix-inversion methods are similar in structure to the implicit function based sensitivity methods in standard traffic assignment literature [102].

shown to be computationally efficient than methods based on matrix-inversion.

The use of sensitivity analysis for network contraction is not new. Hearn [108] provided a transfer decomposition approach for subnetwork contraction. Lately, bush-based sensitivity methods have been used for network contraction and it has been shown that such methods can iteratively improve the computational performance of solving traffic assignment [109, 110]. In this chapter, we exploit the acyclic network structure of MLs and extend the algorithms in the literature for UER sensitivity that scale well for large networks.

6.3 Preliminaries

The notation in this chapter broadly follows the notations in earlier chapter; however, some variables are shortented for easier reading while some are restated for independent reading.

In this section, we introduce the notations for UER models and propose a gradient projection algorithm for solving the equilibrium. All the assumptions are marked as A# and ideas for future work are marked as FW#.

6.3.1 Supply-side uncertainty

Let $G = (N, A)$ denote a network with N as the set of nodes and A as the set of links. Let $\Gamma(i)$ and $\Gamma^{-1}(i)$ denote the sets of downstream and upstream nodes of node i, respectively. Let $Z \subseteq N$ denote the set of all nodes where trips begin or end. For each $(u, v) \in Z^2$, let d_{uv} denote the number of travelers from origin u to destination v.

Due to travel time and toll uncertainty, each link $(i, j) \in A$ exists in one of the multiple cost states called *link-states*. Let S_{ij} represent the set of all link-states for link $(i, j) \in A$ and $S = \bigcup_{(i,j)\in A} S_{ij}$ represent the set of all link-states in the network. On traversing a link each traveler incurs a cost depending on the link-state. On managed lane networks, link-states model variable toll on MLs or variable travel time on GPLs. We assume that link travel times and tolls can be combined linearly as a generalized cost and the parameters in the generalized cost expression (like the value of time) are homogeneous across the population (A#1). This assumption is made to simplify the analysis and considering heterogeneity of parameters is left as part of the future work (FW#1). A link-state $s \in S$ is associated with

a unique link. Let $\texttt{link}(s)$ be the function returning the link associated with state s, and $\texttt{tail}(s)$ and $\texttt{head}(s)$ be the functions returning the tail and head nodes of $\texttt{link}(s)$.

Let p_s denote the probability that link-state $s \in S$ is realized on $\texttt{link}(s)$. We assume that the number of link-states for each link is finite and the probability of occurrence of a link-state is independent of all other links (A#2). This assumption is commonly made across the UER literature and is reasonable in a static setting where cost interactions between different links (due to factors like queue spillback) are not modeled. Limited cases of correlation between link-state probabilities can also be handled in the same framework [91], but the approach is not discussed here for notational brevity. We also assume that if a traveler revisits a link, the probabilities of link-states are reset (A#3). This is called the full-reset assumption in the literature [20]. Given managed lane networks are acyclic (because of the directional nature of freeways), the full-reset assumption has less significance in this chapter as no link is revisited by any traveler; however, we include it for the completeness of arguments.

Let x_s denote the total vehicular flow using $\texttt{link}(s)$ in state $s \in S$ (x_s may be fractional as we model non-atomic travelers). Let $c_s(x_s)$ denote the generalized cost in link-state s as a function of total flow in the link-state x_s. We compute $c_s(x_s)$ as a linear combination of travel time and toll in link-state s, denoted by $t_s(x_s)$ and $\tau_s(x_s)$, respectively. Similar to the assumptions on link costs for sensitivity analysis of standard traffic assignment, we assume that functions $c_s(\cdot)$ are separable by link-state flows, and are positive, strictly increasing, continuous, and differentiable functions of flow for all $s \in S$ (A#4). This assumption will later help establish uniqueness properties of equilibrium and sensitivity parameters.

6.3.2 Information provision and routing of single traveler

Travelers on managed lane networks receive real-time information about tolls and travel time through various information sources like variable message signs and/or mobile applications and choose links minimizing their expected costs. We assume that upon arrival at a node $i \in N$ a traveler learns the realization of link-states on all its downstream links $(i, j) \in A$ and only those links (A#5). Assumption A#5 also makes the implicit assumption that while traversing a link the travel times will not change (referred as the temporal-

dependence assumption in Waller and Ziliaskopoulos [19]).

Given the assumption that link-states are uncorrelated across different links (assumption A#2) and the full-reset assumption (assumption A#3), assumption A#5 does not sacrifice generality as observing any link other than the ones immediately downstream adds no more information at the current node for a traveler minimizing their expected cost. This structure for information provision also permits the use of standard MDP algorithms to determine least-expected-cost strategies for a traveler.

Let $\theta \in \Theta_i = \times_{j \in \Gamma(i)} S_{ij}$ denote the information vector received at node i, where Θ_i is the set of all possible information that can be received at node i. Let θ_{ij} denote the link-state of link (i, j) under information θ at node i. Let q_i^θ denote the probability of receiving message $\theta \in \Theta_i$ at node i. Using assumption A#2, we get $q_i^\theta = \Pi_{j \in \Gamma(i)} p_{\theta_{ij}}$. We define (i, θ) tuple as a *node-state* at node i. Each node-state corresponds to a decision point for a traveler. Additionally, let $\Phi = \{(i, \theta) : i \in N, \theta \in \Theta_i\}$ denote the set of all node-states.

As an example, consider the network shown in Figure 6.2 where links $(1, 2)$ and $(2, 3)$ form the managed lane while the link $(1, 3)$ forms the general purpose lane (GPL). Link $(1, 3)$ exists in two possible states s_2 and s_3 with occurence probabilities as 0.6 and 0.4, respectively, while the other two links only exist in one link-state. Two node-states $(1, \theta)$ are possible at node 1, one with $\theta_{13} = s_2$ while the other with $\theta_{13} = s_3$. Node-states and link-states for this network can be visualized using a network transformation as shown in Figure 6.3.

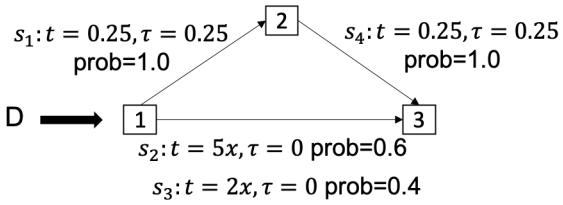


Figure 6.2: An example managed lane network where links $(1, 2)$ *and* $(2, 3)$ *are managed lanes with fixed toll, while link* $(1, 3)$ *is regular lane with two link states under variable travel times*

At each node-state $(i, \theta) \in \Phi$, a traveler chooses a downstream node $j \in \Gamma(i)$ to traverse the network and reach their destination. This decision making for a traveler is captured by a *policy* that determines an action in any node-state. We define a policy π :

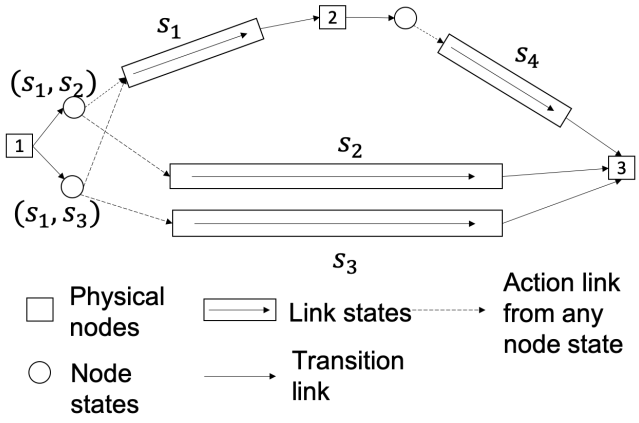


Figure 6.3: Network transformation show the node-states and link-states for the network in Figure 6.2

$\Phi \to N$ as a function that maps each node-state to a downstream node or a terminal node if the node is a destination. We define two properties of a policy. First, a policy *terminates* at i if $\pi(i, \theta) = i$ for all $\theta \in \Theta_i$. This is typically the destination node for any policy. Let $\texttt{dest}(\pi) \in N$ denote the node where the policy π terminates. Second, a policy is defined as *non-waiting* if for all $i \in N \setminus \{\texttt{dest}(\pi)\}$, $\pi(i, \theta) \neq i$. Since each traveler must end their travel at their destination and should not wait idly at intermediate nodes except the destination, we only consider non-waiting policies that terminate at the destination node (A#6). Let $\hat{\Pi}_v$ denote the non-waiting policies terminating at destination $v \in Z$ and $\hat{\Pi} = \bigcup_{v \in Z} \hat{\Pi}_v$ be the set of all policies.[3]

If the link generalized costs are held constant, the problem of finding a policy that minimizes the expected cost for any traveler terminating at destination $v \in Z$ is called the online shortest path (OSP) problem which has been extensively studied in the literature [19, 31]. The OSP problem can be formulated as an MDP with the state space as Φ, the action space as N, and the cost of choosing action $j \in \Gamma(i)$ in state $(i, \theta) \in \Theta_i$ as $c_{\theta_{ij}}$. Since the state and action spaces are finite and the costs are positive (and thus bounded from below), there exists a deterministic policy solving the MDP associated with the OSP problem [111].

[3]In contrast to the formulations in Li et al. [107], we consider deterministic policies. It is possible to write the formulations in the following sections in terms of a stochastic policy $\pi_{\text{stoch}} : \Phi \times A \to [0, 1]$ which determines the probability of choosing link $(i, j) \in A$ for any node state $(i, \theta) \in \Phi$. However, we prefer a deterministic format for drawing parallels from traditional user equilibrium where a path connecting origin to destination generates deterministic link choice at nodes along the path.

Let C_i^π be the the expected travel cost from node i to destination v following a policy $\pi \in \hat{\Pi}_v$. Define $\rho_{i,s}^\pi$ as the probability of leaving node $i = \texttt{tail}(s)$ via link-state s while following policy π, evaluated using Equation (6.1). Because we consider non-waiting policies, $\sum_{j\in\Gamma(i)} \sum_{s\in S_{ij}} \rho_{i,s}^\pi = 1$ for all $i \in N$ and $\pi \in \hat{\Pi}$.

$$\rho_{i,s}^\pi = \sum_{\theta\in\Theta_i \text{ s.t. } \pi(i,\theta)=j,\theta_{ij}=s} q_i^\theta \tag{6.1}$$

The expected cost from each node i to the destination v following a policy $\pi \in \hat{\Pi}_v$ can be calculated using following recursive relation, referred as Bellmann equations.

$$C_v^\pi = 0 \tag{6.2}$$

$$C_i^\pi = \sum_{j\in\Gamma(i)} \sum_{s\in S_{ij}} \rho_{i,s}^\pi \left(t_s + C_j^\pi\right) \tag{6.3}$$

Due to the acyclic structure of the managed lane networks, Bellman equations can be reduced to a simpler expression[4]. Define $\boldsymbol{C}^\pi \in \mathbb{R}_+^{|N|\times 1}$ as a vector of expected costs C_i^π with nodes arranged in the topological order. Let $o(i)$ be the order of node $i \in N$ given a topological ordering. Furthermore, define $\mathbf{c}$ a vector of c_s in some order of states such that for any two nodes $i \in N$ and $l \in N$, if $o(i) < o(l)$, then all link-states $s \in S$ with $\texttt{tail}(s) = l$ are listed after all link-state $s' \in S$ with $\texttt{tail}(s') = i$.

We argue that vectors $\boldsymbol{C}^\pi$ and $\mathbf{c}$ are related using the following expression:

$$\boldsymbol{C}^\pi = \boldsymbol{F}_\pi \mathbf{c} \tag{6.4}$$

where $\boldsymbol{F}_\pi \in \mathbb{R}_+^{|N|\times|S|}$ is an upper trapezoidal matrix. Algorithm 7 shows how to estimate the elements of $\boldsymbol{F}_\pi$ using a single pass through the network in reverse topological order.

For example, for the network in Figure 6.2, if policy π_1 selects link $(1, 3)$ in both node-states at node 1 and policy π_2 selects link $(1, 3)$ only if it is observed in state s_3, then $\boldsymbol{F}_{\pi_1}$ and $\boldsymbol{F}_{\pi_2}$ matrices are given by Equation (6.5). It is easy to verify that Algorithm 7

[4]We can derive similar expression for policies that have finite number of cycles assuming full-reset; however, in this chapter, we only focus on policies with no cycles

Algorithm 7 Algorithm for determining $\boldsymbol{F}_\pi$ matrix

Input: Policy π terminating at $\texttt{dest}(\pi)$, probabilities $\rho^\pi_{i,s}$ for all $i \in N, s \in S$, and a topological ordering $o(\cdot)$ for the ML network

Initialize $f_\pi(i,s) = 0$ for all $i \in N, s \in S$
for node i in reverse topological order starting with $i = \texttt{dest}(v)$ **do**
  for $j \in \Gamma(i)$ **do**
    sum $= 0$
    for $s \in S_{ij}$ **do**
      $f_\pi(i,s) = \rho^\pi_{i,s}$
      sum $=$ sum $+ \rho^\pi_{i,s}$
    end for
    for all $s' \in S$ **do**
      if $f_\pi(i,s') > 0$ **then**
        $f_\pi(i,s') = f_\pi(i,s') +$ sum $\times f_\pi(j,s')$
      end if
    end for
  end for
end for

indeed relates the node expected costs with that of link-state costs. Matrix $\boldsymbol{F}_\pi$ generalizes the $\rho^\pi_{i,s}$ for any combination of i and s and denotes the cumulative probability for all "routes" connecting i and s.

$$\boldsymbol{F}_{\pi_1} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} s_1 & s_2 & s_3 & s_4 \end{array} \\ \begin{pmatrix} 0 & 0.6 & 0.4 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}, \boldsymbol{F}_{\pi_2} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} s_1 & s_2 & s_3 & s_4 \end{array} \\ \begin{pmatrix} 0.6 & 0 & 0.4 & 0.6 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \tag{6.5}$$

An alternate way to visualize the elements of $\boldsymbol{F}_\pi$ matrix is if we remove the node state in the transformation in Figure 6.3, resulting in a revised transformation as in Figure 6.4. This revised transformation forms a directed acyclic graph, which we call *policy-probability* graph, rooted at destination $\texttt{dest}(\pi) = v$. We next define the notion of physical path in this graph and relate element of $\boldsymbol{F}_\pi$ matrix with the product of probabilities on arcs along a path.

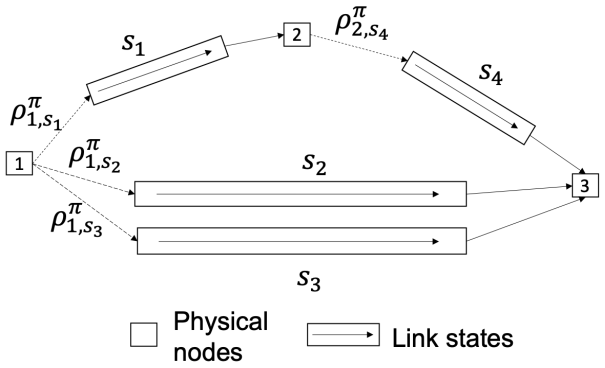


Figure 6.4: Network transformation show the node to link-state connection with probabilities on each arc expressed in terms of ρ variable.

Define a directed path as a sequence of physical nodes and link-states connecting two nodes. For example, $[1, s_1, 2, s_4, 3]$ is a path connecting 1 and 3 in Figure 6.4. Let ζ_{mn} be all directed paths connecting nodes $m \in N$ and $n \in N$. For any directed path $z \in \zeta_{mn}$, we define $\chi^{\pi,z}_{m\to n}$ as the probability that travelers from node m will pass through n via path z if they were following policy π. The value of $\chi^{\pi,z}_{m\to n}$ is the product of probabilities on each node to link-state arc along path z.

The element $f_\pi(i, s)$ denotes the probability that cost from the link-state s will affect an upstream node i, which is equal to the probability that any directed path connecting nodes i to v contains state s, weighted by the cumulative probability of each path. That is,

$$f_\pi(i, s) = \sum_{z\in\zeta_{iv}: s\in z} \chi^{\pi,z}_{i\to \texttt{dest}(\pi)}. \tag{6.6}$$

For example, for the network in Figure 6.4, $f_\pi(1, s_4) = \chi^{\pi,[1,s_1,2,s_4,3]}_{1\to 3} = \rho^{\pi}_{1,s_1}\rho^{\pi}_{2,s_4}$ for any $\pi \in \hat{\Pi}_3$. Additionally, $\rho^{\pi}_{2,s_4} = 1$ for all $\pi \in \hat{\Pi}_3$ as s_4 is the only choice at node 2. Thus, $f_\pi(1, s_4) = \rho^{\pi}_{1,s_1}$. For π_1 and π_2 defined earlier, ρ^{π}_{1,s_1} equals 0 and 0.6 respectively, resulting in the matrix elements in Equation (6.5).

6.3.3 Multiple travelers and flow variables

Under the presence of multiple travelers, a traveler's choice of policy depends on the policies chosen by other travelers. Since each policy is a solution for an MDP, this is referred as an MDP congestion game [107]. Similar to the earlier work on congestion games we

assume that travelers are non-atomic in nature (A#7) and thus the congestion game is a mean-field game, that is number of travelers using any policy is a continuous variable.

The Wardrop-type equilibrium for the MDP congestion game, referred as UER, defines the steady-state equilibrium for the game. At UER, all used policies $\pi \in \hat{\Pi}_v$ for travelers from origin u to destination v have equal and minimal expected costs [16].

We introduce x_s^π as the flow on link-state s following policy π towards $\texttt{dest}(\pi)$. Total flow on each link state is given by $x_s = \sum_{\pi \in \hat{\Pi}} x_s^\pi$. Furthermore, let y_u^π be the flow originating from node $u \in Z$ following policy π headed towards destination $\texttt{dest}(\pi)$. Flow on each policy is non-negative, $y_u^\pi \geq 0$, for all origins $u \in Z$ and policies π, and conservation of flow requires that the total flow on all policies between the origin and destination equals the demand, that is $d_{uv} = \sum_{\pi \in \hat{\Pi}_v} y_u^\pi$.

We define $\mathbf{x}^\pi \in \mathbb{R}_+^{|S|}$ be a vector with elements x_s^π in the same order as the $\mathbf{c}$ vector, and define $\mathbf{y}^\pi \in \mathbb{R}_+^{|N|}$ as a vector of y_i^π for all nodes $i \in N$ (we set $y_i^\pi = 0$ if $i \notin Z$). Then, $\mathbf{x}^\pi$ and $\mathbf{y}^\pi$ satisfy the relation in Equation (6.7).

$$\mathbf{x}^\pi = \boldsymbol{F}_\pi^\intercal \mathbf{y}^\pi \tag{6.7}$$

This relation is easy to verify. Flow through a link-state is a weighted sum of flows on paths in the corresponding policy-probability graph (refer Figure 6.4). The flow from any node i following policy π that passes through link-state s is equal to the flow from the node following that policy times the total probability that any directed path from i to $\texttt{dest}(\pi)$ will pass through s (which is same as $f_\pi(i, s)$). Summing that across all nodes, we get $x_s^\pi = \sum_{i \in N} f_\pi(i, s) y_i^\pi$, which is an expanded form for Equation (6.7). Because demand only originates at nodes in set Z, $y_i^\pi = 0$ for all $i \notin Z$. Thus, we can rewrite Equation (6.7) as $x_s^\pi = \sum_{i \in Z} f_\pi(i, s) y_i^\pi$.

6.3.4 UER convex program

Define $\mathbf{y}$ as a vector of all y_u^π for all $u \in Z$, $\pi \in \hat{\Pi}_v$, and $v \in Z$. At UER, all used policies between an origin and a destination have equal and minimal expected costs, that is:

$$y_u^\pi > 0 \Rightarrow C_u^\pi(\mathbf{y}) = \min_{\bar{\pi} \in \hat{\Pi}_v} C_u^{\bar{\pi}}(\mathbf{y}) \qquad \forall v \in Z, \pi \in \hat{\Pi}_v \tag{6.8}$$

For finding the UER policy flows satisfying this principle, we define a convex program as follows:

$$\min_{\mathbf{y},\mathbf{x}^\pi,\mathbf{x}} Z(\mathbf{y}) = \sum_{s \in S} \int_0^{x_s} c_s(w) dw \tag{6.9}$$

$$\text{s.t. } d_{uv} = \sum_{\pi \in \hat{\Pi}_v} y_u^\pi \qquad \forall (u,v) \in Z^2 \tag{6.10}$$

$$x_s^\pi = \sum_{u \in Z} f_\pi(u,s) y_u^\pi \qquad \forall s \in S, \pi \in \hat{\Pi} \tag{6.11}$$

$$x_s = \sum_{\pi \in \hat{\Pi}} x_s^\pi \qquad \forall s \in S \tag{6.12}$$

$$y_u^\pi \geq 0 \qquad \forall \pi \in \hat{\Pi}, u \in Z \tag{6.13}$$

Proposition 4. *The optimal solutions to the convex program* (6.9)–(6.13) *correspond exactly to the policy flows satisfying the UER definition in Equation* (6.8)*.*

Proof. The argument is identical to the proof in Rambha et al. [16] and Unnikrishnan and Waller [15], where we can that the KKT conditions of the convex program are equivalent to the UER principle. □

We next present a gradient-projection algorithm in the space of policies to solve the UER problem.

6.3.5 Gradient projection algorithm to solve UER

Similar to the gradient projection algorithms for the traffic assignment problem [112], we propose an algorithm that works directly in the space of policies and shifts travelers

among policies between an origin and a destination until the UER condition is satisfied.

First, we define a set $\hat{\Pi}^{uv} = \{\pi \in \hat{\Pi}_v \mid y_u^\pi > 0\}$ as the set of all used policies between an origin-destination pair $(u, v) \in Z^2$. The gradient projection algorithm finds the shortest policy connecting u and v and adds it to $\hat{\Pi}^{uv}$ if it doesn't already exist. Then, it calculates the shift of flows among the set of used policies that minimizes the Beckmann-like objective function in Equation (6.9) and projects the shift onto the space of feasible policy flows.

Rewriting the convex-UER formulation in terms of policy flows alone, we obtain:

$$\min_{\mathbf{y}} Z(\mathbf{y}) = \sum_{s \in S} \int_0^{\sum_{(u,v)\in Z^2} \sum_{\pi \in \hat{\Pi}_v} f_\pi(u,s) y_u^\pi} c_s(w) dw \tag{6.14}$$

$$\text{s.t. } d_{uv} = \sum_{\pi \in \hat{\Pi}_v} y_u^\pi \qquad \forall (u, v) \in Z^2, \tag{6.15}$$

$$y_u^\pi \geq 0 \qquad \forall \pi \in \hat{\Pi}, u \in Z. \tag{6.16}$$

Define a *basic* policy $\pi_{uv}^* \in \hat{\Pi}_{uv}$ for an OD pair $(u, v) \in Z^2$ to be a policy with minimum expected cost to destination v from node u. All the other policies are called *nonbasic* policies. We can eliminate the basic policy flow variable for each OD pair by expressing it in terms of nonbasic policy flows using the demand conservation constraint. Let $\bar{y}_u^\pi$ be the variable representing flows on all nonbasic policies $\pi \in \hat{\Pi}_{uv} \setminus \{\pi_{uv}^*\}$ and $\bar{\mathbf{y}}$ be a vector of $\bar{y}_u^\pi$. We can express the flows on the basic policy using demand conservation as follows:

$$y_u^{\pi_{uv}^*} = d_{uv} - \sum_{\pi \in \hat{\Pi}_{uv}: \pi \neq \pi_{uv}^*} \bar{y}_u^\pi \qquad \forall (u, v) \in Z^2. \tag{6.17}$$

We then define a modified objective after eliminating basic policy flow $y_u^{\pi_{uv}^*}$ for each OD pair.

$$\min_{\bar{\mathbf{y}}} \bar{Z}(\bar{\mathbf{y}}) = \sum_{s \in S} \int_0^{\sum_{(u,v)\in Z^2} \left[\left(\sum_{\pi \in \hat{\Pi}_v} f_\pi(u,s) \bar{y}_u^\pi\right) + f_{\pi_{uv}^*}(u,s)\left(d_{uv} - \sum_{\pi \in \hat{\Pi}_{uv}; \pi \neq \pi_{uv}^*} \bar{y}_u^\pi\right)\right]} c_s(w) dw \tag{6.18}$$

$$\bar{y}_u^\pi \geq 0 \qquad \forall \pi \in \hat{\Pi}; \pi \neq \pi_{uv}^*, u \in Z \tag{6.19}$$

It is easy to verify that the convex programs in Equations (6.14)–(6.16) and Equations (6.18)–(6.19) are identical. Because the only constraint for the latter convex program is the non-negativity constraint on policy flows, we can easily apply gradient projection algorithms for computing an optimal solution. A gradient projection algorithm involves finding the direction of steepest descent, which requires computing first order derivative of $\bar{Z}$.

$$\frac{\partial \bar{Z}}{\partial \bar{y}_u^\pi} = \sum_{s \in S} c_s(x_s(\bar{\mathbf{y}})) \left[f_\pi(u,s) - f_{\pi^*_{uv}}(u,s) \right] \tag{6.20}$$

$$= C_u^\pi(\bar{\mathbf{y}}) - C_u^{\pi^*_{uv}}(\bar{\mathbf{y}}) \tag{6.21}$$

The gradient of objective wrt to a policy flow $\bar{y}_u^v$ is simply the difference between the cost from node u to $\texttt{dest}(\pi)$ following policies π and π^*_{uv}. Since basic policy has the least expected cost, the first-order derivative is always positive and the descent direction can only reduce flows from the nonbasic policies. Additionally, since we can compute the second-order derivative for the objective, we can improve the search direction by dividing the first-order derivative with the second-order derivative, which is computed as follows:

$$\frac{\partial^2 \bar{Z}}{\partial \left(\bar{y}_u^\pi\right)^2} = \frac{\partial}{\partial \bar{y}_u^\pi} \sum_{s \in S} \left(c_s(x_s) \left[f_\pi(u,s) - f_{\pi^*_{uv}}(u,s) \right] \right) \tag{6.22}$$

$$= \sum_{s \in S} \left[f_\pi(u,s) - f_{\pi^*_{uv}}(u,s) \right] c_s'(x_s(\bar{\mathbf{y}})) \frac{\partial x_s(\bar{\mathbf{y}})}{\partial y_u^\pi} \tag{6.23}$$

$$= \sum_{s \in S} \left[f_\pi(u,s) - f_{\pi^*_{uv}}(u,s) \right]^2 c_s'(x_s(\bar{\mathbf{y}})) \tag{6.24}$$

Given these derivatives, we can write the gradient projection algorithm as in Algorithm 8.

Algorithm 8 Policy-based gradient projection algorithm for solving UER

Initialize $\hat{\Pi}_{uv} = \text{NULL}$ for all $(u, v) \in Z^2$. Set iteration number $n = 0$. Set $x_s\rfloor_n = 0$ for all $s \in S$. Set $t_s = t_s(0)$ for all $s \in S$. Set GAP $\leftarrow \infty$

while GAP $> \epsilon$ **do**
  for Each destination $v \in Z$ **do**
    Find shortest policy π_v^* towards destination v using `TD-OSP` algorithm
    for Each origin $u \in Z$ with $d_{uv} > 0$ **do**
      Shortest policy $\pi_{uv}^* = \pi_v^*$
      if $\pi_v^* \notin \hat{\Pi}_{uv}$ **then** $\hat{\Pi}_{uv} \leftarrow \hat{\Pi}_{uv} \cup \{\pi_v^*\}$
      end if
      if $|\hat{\Pi}_{uv}| = 1$ **then**
        Set $y_u^{\pi_{uv}^*}\rfloor_{n+1} \leftarrow d_{uv}$
      else $|\hat{\Pi}_{uv}| > 1$
        Set total flow to shift towards basic policy as zero: $\Delta y^* \leftarrow 0$
        for $\pi \in \hat{\Pi}_{uv}$ such that $\pi \neq \pi_{uv}^*$ **do**

$$\Delta y \leftarrow \min \left\{ y_u^{\pi}\rfloor_n, \frac{C_u^{\pi} - C_u^{\pi_{uv}^*}}{\sum_{s \in S} c_s'(x_s\rfloor_n)(f_{\pi}(u,s) - f_{\pi_{uv}^*}(u,s))^2} \right\} \tag{6.25}$$

$$y_u^{\pi}\rfloor_{n+1} \leftarrow y_u^{\pi}\rfloor_n - \Delta y \tag{6.26}$$

$$\Delta y^* \leftarrow \Delta y^* + \Delta y \tag{6.27}$$

        end for
      end if
      Set $y_u^{\pi_{uv}^*}\rfloor_{n+1} \leftarrow y_u^{\pi_{uv}^*}\rfloor_n + \Delta y^*$
    end for
  end for
  GAP $\leftarrow \left(\sum_{s \in S} t_s x_s\rfloor_n\right) \left(\sum_{(u,v) \in Z^2} d_{uv} C_u^{\pi_{uv}^*}\right)^{-1} - 1$
  Update $x_s\rfloor_{n+1}$ using $y_u^{\pi}\rfloor_{n+1}$ values
  $n = n + 1$
end while

The algorithm starts with an empty used policy set and incrementally adds policies by generating shortest expected cost policies for each destination in each iteration and adding them to the used policy set if it doesn't already exist. The `TD-OSP` algorithm from Waller and Ziliaskopoulos [19] is used to determine the shortest expected cost policy towards any destination. If the set of used policies is a singleton, all demand is loaded onto the only policy, or else the flow is shifted from every nonbasic policy to the basic policy using the gradient descent update rule (Equation (6.25), where we assume the step size is set to 1). The convergence criterion measured using GAP function is identical to the other relative gap

metrics used in the traffic assignment literature [113]. The comparison of whether a policy is identical to another is done by testing the equivalence of $\boldsymbol{F}_\pi$ matrix associated with the policy.

6.4 Sensitivity analysis model

In this section, we conduct sensitivity of equilibrium costs to changes in demand values between an origin-destination pair used for network contraction. The artificial link models the variation in expected cost as a function of OD demand. For example, for the network in Figure 6.2, we seek parameters of cost function on the artificial link connecting nodes 1 and 3 as a function of demand (Figure 6.5). We seek the derivative of expected cost at equilibrium between nodes u and v as a function of demand between the nodes.

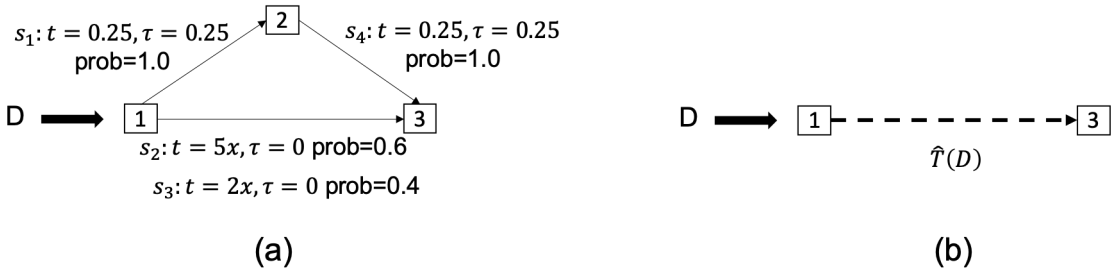


Figure 6.5: (a) Small network and (b) a contracted network with artificial link representing demand as a function of cost

We start with making following additional assumptions. First, for the derivatives to be defined, we assume that the current UER equilibrium solution is non-degenerate in policy-flow space and satisfies "strict-complementarity" in terms of policy flows (A#8). That is, at equilibrium all unused policies between an origin-destination pair have strictly higher costs than the used policies. This assumption is reasonable as the number of points of degeneracy in a network are finite.

Second, we assume that the demand perturbation is small enough that it preserves the set of used policies (A#9). This assumption is commonly made across the sensitivity analysis literature [105]. Similar to the analysis in Lu and Nie [114], we hypothesize that variations in policy-flows are continuous for small changes in demand.

Last, we assume that the expected travel time between an origin-destination (OD)

pair only depends on the demand between that OD pair and not on other demand values (A#10). This is called the separability assumption of OD pairs. This assumption is reasonable if policies between two OD pairs do not overlap significantly. While this assumption may not hold true for general networks, as we see in the experiments conducted in Section 6.5, the error in expected costs with this assumption are not significant.

Our goal is to estimate the derivative of expected cost from a node u towards the destination v with respect to the demand between the two nodes. That is, we seek $\partial C_u^\pi / \partial d_{uv}$. Let $\mathbf{x}^*$ and $\mathbf{y}^*$ be the link-state and policy flows at the current UER solution and the local derivatives are to be estimated at this current solution. Let Π_{uv}^{used} be the set of all used policies between origin-destination pair (u, v) and let $\Pi^{\text{used}} = \bigcup_{(u,v)\in Z^2} \Pi_{uv}^{\text{used}}$ be the set of all used policies. The following conditions hold at UER:

$$C_u^\pi = \sum_{s\in S} f_\pi(u,s) c_s(x_s^*) \qquad \forall \pi \in \Pi_{uv}^{\text{used}}, u \in Z, v \in Z \tag{6.28}$$

$$x_s^* = \sum_{\pi\in\Pi^{\text{used}}} \sum_{i\in Z} f_\pi(i,s) y_i^{\pi,*} \qquad \forall s \in S \tag{6.29}$$

$$\sum_{\pi\in\Pi_{uv}^{\text{used}}} y_u^{\pi,*} = d_{uv} \qquad \forall u \in Z, v \in Z \tag{6.30}$$

We define partial derivatives of node cost-to-go values, link-state flows, and policy flow with respect to d_{uv}. Let $\rho_i^\pi = \frac{\partial C_i^\pi}{\partial d_{uv}}$, $\alpha_s = \frac{\partial x_s^*}{\partial d_{uv}}$, $\beta_i^\pi = \frac{\partial y_i^{\pi,*}}{\partial d_{uv}}$. Because of the separability of the OD pair assumption, we set $\beta_i^\pi = 0$ and $\rho_i^\pi = 0$ for all $i \neq u$ and for all $\pi \notin \Pi_{uv}^{\text{used}}$.

Differentiating the optimality conditions in Equations (6.28)–(6.30) with respect to d_{uv}, we obtain:

$$\rho_u^\pi = \sum_{s\in S} f_\pi(u,s) c_s'(x_s^*) \alpha_s \qquad \forall \pi \in \Pi_{uv}^{\text{used}} \tag{6.31}$$

$$\alpha_s = \sum_{\pi\in\Pi_{uv}^{\text{used}}} \sum_{i\in Z} f_\pi(i,s) \beta_i^\pi \qquad \forall s \in S \tag{6.32}$$

$$\sum_{\pi\in\Pi_{uv}^{\text{used}}} \beta_u^\pi = 1 \tag{6.33}$$

These conditions are the optimality conditions of following convex program.

$$\min_{\boldsymbol{\beta},\boldsymbol{\alpha}} \sum_{s\in S} \int_0^{\alpha_s} (c_s)'_{\rfloor x_s = x_s^*}\, w\; dw \tag{6.34}$$

$$\text{s.t. } 1 = \sum_{\pi\in\Pi_{uv}^{\text{used}}} \beta_u^\pi \tag{6.35}$$

$$\alpha_s = \sum_{\pi\in\Pi_{uv}^{\text{used}}} f_\pi(s,u)\beta_u^\pi \qquad \forall s \in S \tag{6.36}$$

The convex program is identical to the UER problem with $\left(c_{ij}^s(\mathbf{x}^*)\right)' w$ as new linear cost function for each link-state, α_s as the new flow on each link-state and β_u^π as the new flow on each policy π. However, there is no non-negativity constraint. We can solve this convex program using the gradient projection algorithm in Algorithm 8 with the exception that we allow flows to be negative (that is, exclude the min from Equation (6.25)).

6.5 Experiments

In this section, we report experimental results for the algorithms proposed in previous section. First, consider the example network in Figure 6.2. Given simpler network structure, the equilibrium flows on link-states can be solved analytically. Figure 6.6 shows the variation of expected costs between nodes 1 and 3 as a function of demand between the nodes.

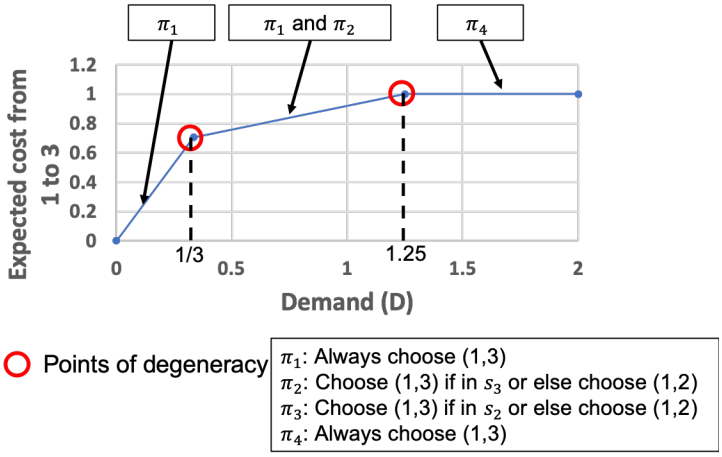


Figure 6.6: Variation of expected cost between nodes 1 and 3 for varying values of demand highlighting a possible set of used policies for each linear region and the points of degeneracy where the expected cost is non-differentiable

Since the cost functions are linear, the variation of expected cost is linear with respect to variation in demand. There are three linear regions with constant values of derivatives based on which policies are being used and solving the UER sensitivity algorithm is able to estimate these derivatives exactly. There are two demand values, $D = 1/3$ and $D = 1.25$, where the costs of unused policies are identical to the equilibrium expected cost between 1 and 3, and the derivative is not defined for those points.

For the real-world experiments, we consider the toll-segment 2 of the North-Tarrant Expressway (NTE) network in Dallas, TX. The network consists of 54 nodes and 67 links. The network was extracted from the original demand model provided by the North Central Texas Council of Government. The free-flow travel time and capacity of each link were used as is and the travel time function was assumed to follow the standard non-linear BPR function. The tolls on managed lanes were as obtained from the model. The supply-side uncertainty in the network was added by creating two additional states on the links on general purpose lanes with revised free-flow travel times as 0.7 and 2 times the true value, and two additional states on the managed lanes with tolls multiplied by 0.7 and 2 times the true value. Figure 6.7 shows the network with the origin and the destination nodes highlighted.

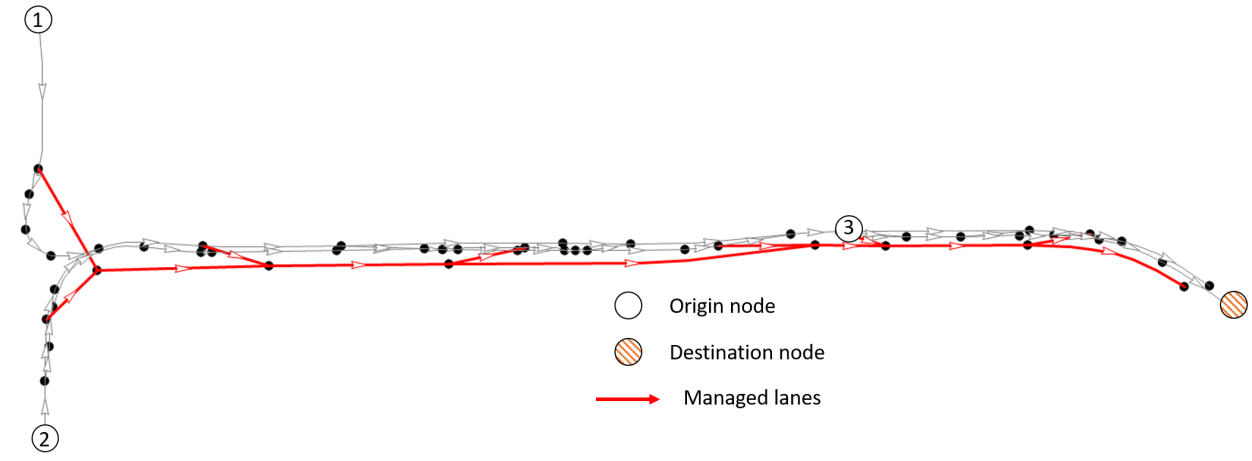


Figure 6.7: Toll segment 2 of the North Tarrant Expressway, Dallas, TX

First, we highlight the computational benefit of the gradient projection algorithm proposed in Section 6.3.5. Figure 6.8 shows the variation in relative gap with computation time for the gradient projection and Frank-Wolfe algorithms.

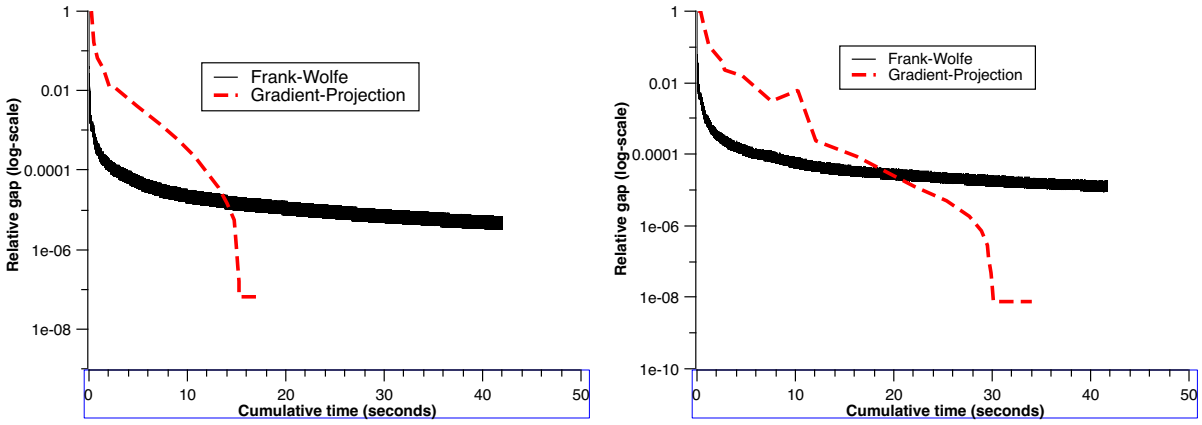


Figure 6.8: Variation of relative gap on the NTE network using two algorithms for low and high demand

For varying levels of demand we find that the gradient projection algorithm is able to obtain solutions with relative gap lower than 1E-8, while the convergence for the Frank-Wolfe algorithm in the earlier literature [15, 16] tails off for latter iterations. This is as expected and relates with the improved decent characteristic of algorithms based on gradient projection in the traffic assignment literature [112].

Second, we study the variation in expected costs (in travel-time units) between each of the three origins and the destination for different demand levels. Seven demand levels were considered where the base demand was multiplied by a factor x, where x belongs to the set $\{0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3\}$. The sensitivity parameters were estimated by first solving the UER at the base demand ($x = 1.0$) and then using the modified-gradient projection algorithm at the obtained solution. The estimated parameters were then used to predict the expected costs for other demand factors.

Figures 6.9(a)–(c) show the predicted expected costs (expressed in minutes) obtained from the sensitivity parameters (dashed lines) and the true expected costs obtained from solving UER for different demand factors (solid line).

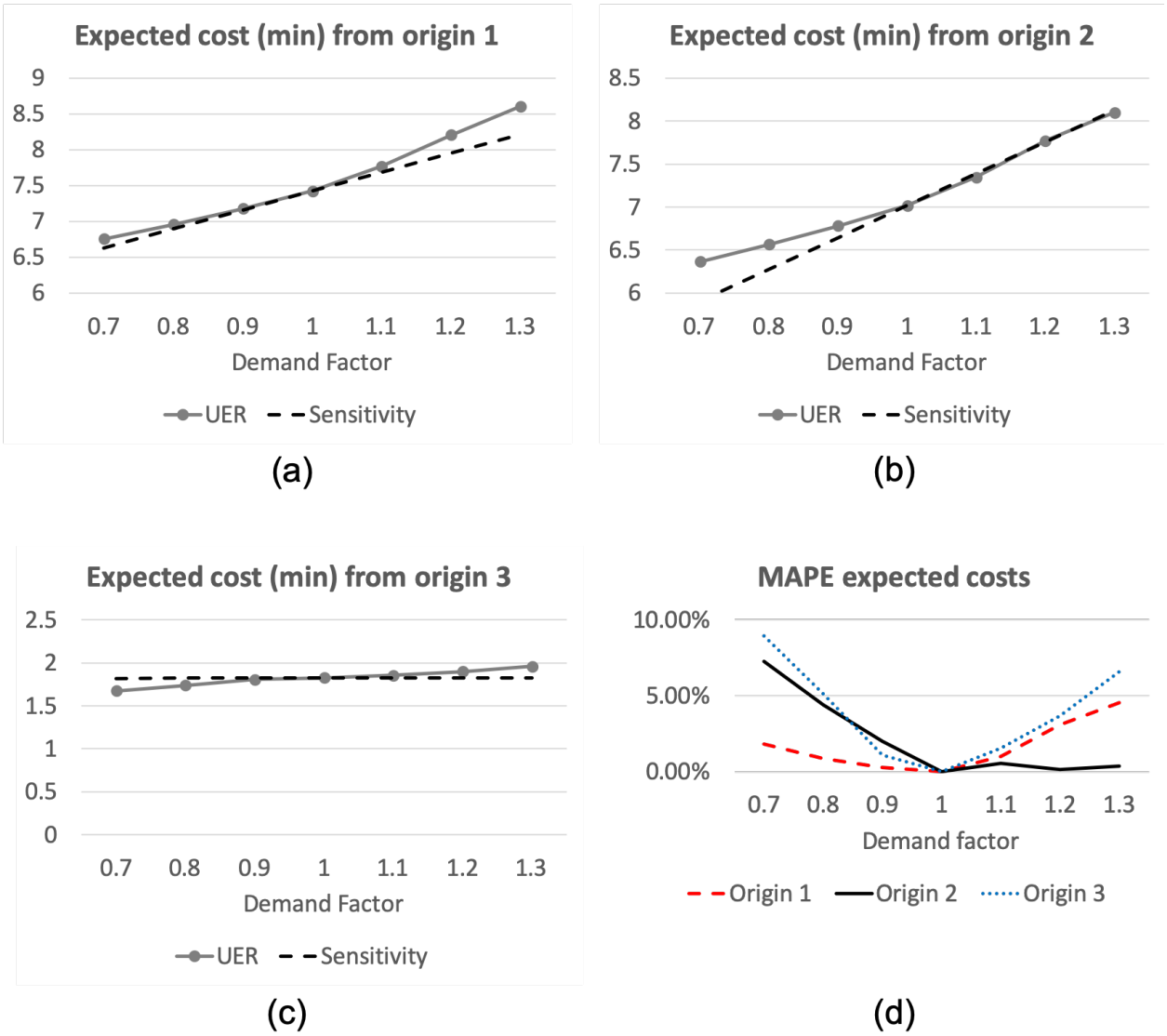


Figure 6.9: Variation in expected costs between (a) origin 1, (b) origin 2, and (c) origin 3 to the destination for varying demand levels. (d) The mean absolute percent error (MAPE) in expected costs for different demand factors for the three origins

As observed, the predicted costs closely mimic the true expected costs obtained from solving UER. The differences are higher for the demand factors farther away from the base demand. This is as expected: the sensitivity parameters were approximated around the base demand and at higher demand perturbation the first-order Taylor series approximation does not hold true. Figure 6.9(d) shows the mean absolute percent error (MAPE) between the true and predicted expected costs for the three origins at varying demand levels. The average MAPE value for the NTE network is 2.94% which is a reasonably low value for transportation

planning purposes. The results suggest that the gradient projection algorithm with linear cost approximation is well suited for approximating true UER expected costs.

6.6 Summary

In this chapter we proposed a convex program to determine the sensitivity of cost parameters at UER with variation in demand between origin-destination pair. The proposed method allows contraction of acyclic express lane corridors into an artificial link whose cost function can be approximated using the first-order Taylor series with derivatives evaluated at a given UER solution. We showed that the mean absolute percent error in expected costs generated using this contraction is up to 10%, or an absolute error of up to a minute, on the NTE corridor in Dallas, TX. We also proposed a gradient-projection based algorithm to solve UER equilibrium solution and show that it is efficient that the Frank-Wolfe algorithm in the literature in generating solutions at low relative gap.

Chapter 7

Conclusion

7.1 Summary

In this dissertation, dynamic pricing and long-term planning models are studied for managed lanes with multiple locations of entrances and exits. These lanes provide reliable travel time in exchange for a toll that varies dynamically based on the congestion pattern. Under the presence of real-time toll and travel time information (provided through variable message signs) and the historic information obtained via past experiences, travelers choose between managed lanes or general purpose lanes and can adapt their lane choices *en route* based on the received information.

Three component models of managed lanes were studied: a single-driver behavior model that explains adaptive lane choice of travelers in presence of real-time and historic information; a dynamic pricing model based on Markov decision process that incorporates adaptive driver behavior for determining tolls that perform better than the existing heuristics; and a user equilibrium with recourse model for express lanes for long-term prediction of traffic under the presence of toll and travel-time uncertainty. The primary finding is that adaptive driver behavior impacts the pricing and planning of express lanes.

For the single-driver behavior model, we considered that a traveler receives real-time information about the tolls and travel times upon arrival at each diverge node and makes a dynamic lane choice decision that minimizes the total expected cost. We formulated the online route choice model as a Markov decision process and solved it using a backward recursion algorithm for acyclic networks. The model was compared against four other routing models including a binary logit model, a model based on decision routes, a model that chooses paths a priori, and a model with routes chosen randomly. We also modeled irrational driver behavior with parameters like driver's inclination towards making optimal lane choices and their preference for certain lanes. The findings show that the expected costs from the routes

chosen using the decision route model are close to the optimal cost with an average percent error of 0.93%. The binary logit model is shown to have a high average error of 50% in the expected cost when a driver is assumed to behave rationally, but the same model shows optimal prediction for certain irrational driver behaviors.

For the dynamic pricing models, we used the decision route model to explain lane choices where travelers make online decisions at each diverge point considering all routes on a managed lane network. We formulated the problem as a Markov decision process and solved it using three algorithms: (a) the VFA algorithm with different initializations, (b) a multiagent reinforcement learning algorithm (SparseV) for decentralized tolling at each toll gantry, and (c) Deep-RL algorithms using neural networks to approximate toll policies. While VFA and SparseV algorithms assume full observability of traffic states, Deep-RL algorithms do not.

Both VFA and SparseV were shown to outperform a myopic revenue policy which maximizes the revenue only at the current timestep and the feedback-control heuristics based on density measurements. We show the revenue-maximizing optimal policies follow the "jam-and-harvest" behavior where the GPLs are pushed towards congestion in the earlier time steps to generate higher revenue in the later time steps, a characteristic not observed for the policies minimizing TSTT.

Under settings of partial observability, Deep-RL algorithms were shown to outperform the feedback-control heuristic by generating up to 10% higher revenue and up to 9% lower delays. These algorithms relax assumptions in the literature by considering multiple origins and destinations, multiple access locations to the managed lane, *en route* diversion of travelers, and stochastic demand and observations. We also proposed reward shaping methods for the pricing model to overcome undesired behavior of toll policies, like the jam-and-harvest behavior of revenue-maximizing policies. Additionally, we tested transferability of the algorithm trained on one set of inputs for new input distributions and offered recommendations on real-time implementations of Deep-RL algorithms. Deep-RL algorithms also outperformed VFA and SparseV methods for the revenue maximization objective generating up to 11.85% higher revenue on an average.

For the long-term planning models of express lanes, we developed a static M-UER

model where travelers are expected to adapt their routes based on the available real-time information. M-UER was formulated as a convex program. The tests conducted on the North Tarrant Expressway network in Dallas, TX showed that ignoring adaptive driver behavior can lead to differences in link flow predictions in the network and underestimation of total system travel cost by 1.6%. We also proposed a gradient-projection algorithm for solving UER on acyclic express lane networks which is shown to be efficient than the existing link-state-based algorithms. Additionally, we analyzed sensitivity of expected costs between two nodes with respect to the demand between the nodes, while assuming that the flow redistributes to maintain the equilibrium. We derived a convex program for estimating the sensitivity parameters and adapted the gradient-projection algorithm developed earlier for an efficient computation of these parameters. This sensitivity analysis allows contracting a complex express lane network as a single directed link. Experiments on real-world networks showed that the proposed network contraction approximates the true variation of expected costs between two nodes very well generating less than 2.5% error in costs on an average. This contraction allows integration of express lanes in existing transportation planning models for large-scale networks while incorporating the adaptive driver behavior in response to network stochasticity.

7.2 Future work

The models developed in this dissertation motivate several topics for future work. The single-driver behavior model in Chapter 2 can be extended for comparing the proposed models when the link travel time and tolls are correlated. Additionally, other criteria involved in the route choice decisions of travelers on managed lanes like reliability can be considered in the generalized cost definition. The model can also be extended for other tolling options discussed in Section 2.2. Route choice data collected from the field can be used to train the parameters of the model which will be highly relevant to companies with current ML installations. This model can form a basis for training dynamic discrete choice models for understanding the route choice behavior of a population based on an individual's demographic or trip type attributes. Last, advanced reinforcement learning algorithms can be used for learning the probability distributions on the fly, which can be useful for efficient

route guidance for navigation apps.

For the dynamic pricing model in Chapters 3 and 4, some future work ideas are mentioned inline (marked as FW#). Other topics for future work include the following. First, for the VFA method in Chapter 3 a weighted aggregation scheme can be designed which uses lower aggregation levels in the first few iterations to achieve faster learning of good value function estimates and switches to higher aggregation levels later. Such a strategy can improve the performance of VFA algorithm for larger networks. Additionally, there are other factors influencing the relative values of states within each time step which can be used to improve the initializations and the convergence characteristics. Second, the choice of traffic flow model is critical to the performance of Deep-RL algorithms in Chapter 4. The macroscopic multiclass cell transmission model does not capture the impacts of lane changes and the second-order stop-and-go waves. Future work can be devoted to developing efficient Deep-RL algorithms using microscopic simulation models and on testing the transferability of algorithms trained on a macroscopic scale to microscopic scales. Third, we only considered loop detector density measurements in the simulations. Other types of observations like speeds, toll-tag readings, and measurements using Lagrangian sensors like GPS devices on vehicles require redefining the POMDP to handle such measurements and can be looked into as part of the future work. Fourth, for real-time implementation of Deep-RL algorithms, the minimum speed limit constraint on ML (constraint 2 defined in Section 4.2.4) should be satisfied throughout the learning phase, which requires analysis of constrained policy optimization methods like in Achiam et al. [115]. Last, the future work should also analyze the equity impacts of the tolls generated by Deep-RL across multiple vehicle classes and investigate if generating equitable toll policies can be included as part of the Deep-RL problem.

The long-term planning models for managed lanes in Chapters 5 and 6 motivate following study topics for the future. First, methodology in Chapter 5 can be extended to continuous VOT distributions by developing a continuous variational inequality and to the cases where the correlation between link travel time and toll values is considered. Second, using the network contraction method proposed in Chapter 6, we can integrate the UER models with the traditional user-equilibrium models. This integration will require an iterative

interaction between the two models where the output from one is an input to the other. Third, there is a need for improving the assumptions on separability of OD pairs made for the estimation of derivatives. Methods from standard traffic assignment literature can be extended [105], though we expect a trade off between the computational efficiency in determining the derivatives and the accuracy of their values. Fourth, the sensitivity analysis method can be used for bi-level network design problems involving UER, where the second-level requires determining solutions to UER for revised parameters. Last, extending UER models to dynamic settings, such as in Gao [17], while developing computationally-tractable models for large-scale managed lane corridors is also an important area for future work.

In addition to the specific future topics for each chapter, there is need for validation of the models developed in this dissertation against field data. Furthermore there is a need for integrating the research on decision making under stress with the models for lane choice. The assumption that travelers seek lane choices minimizing their expected costs is more suitable for connected/autonomous vehicles or where a phone application make the decision on driver's behalf. However, for human drivers, there might be other factors influencing their behavior, one of them being the stress of making a decision right at the diverge location given the information received so far. Future lane choice models should factor this element of stress with decision making. In addition, there is a need for research looking into the social-equity concerns with express lanes and analyzing alternatives to collecting dynamic tolls such as offering driver discounts or a credit-based congestion pricing mechanism [116].

Bibliography

[1] Dimitra Michalaka, Jie Lu, and Yafeng Yin. Fine-tuning pricing algorithms for high-occupancy/toll (HOT) lanes. In *Transportation Research Board 92nd Annual Meeting*, number 13-3992, 2013.

[2] David Schrank, Bill Eisele, and Tim Lomax. TTI 2012 Urban mobility report. *Texas A&M Transportation Institute. The Texas A&M University System*, 2012.

[3] OECD. Improving reliability on surface transport networks. http://www.itf-oecd.org/sites/default/files/docs/10reliability.pdf, Accessed July, 2017.

[4] FHWA. Managed lanes: A primer. *Federal Highway Administration, US Dept. of Transp., Washington DC, USA*, 2013.

[5] n.a. Managed Lanes Project Database. https://managedlanes.wordpress.com/category/projects/, 2020. Last Accessed: January 20, 2020.

[6] LBJ. LBJ express FAQs. http://www.lbjtexpress.com/faq-page/t74n1302, 2016. Last Accessed: June 20, 2019.

[7] NCHRP. Introducing the NCHRP 15-49 implementation guide. In *15th International Conference on Managed Lanes*, number S-13, 2016.

[8] Luz Lazo. Here's a look at who's using Northern Virginia's 495 and 95 Express Lanes. https://www.washingtonpost.com/transportation/2018/09/20/heres-look-whos-using-northern-virginias-express-lanes/, 2018. Accessed January, 2020.

[9] Linda Rosendorf. Virginia's 'Lexus lanes' deserve their nickname. https://www.washingtonpost.com/opinions/virginias-lexus-lanes-deserve-their-nickname/2018/09/26/4eeb182e-c101-11e8-9f4f-a1b7af255aa5_story.html, 2018. Accessed January, 2020.

[10] Feng Zhu and Satish V Ukkusuri. A reinforcement learning approach for distance-based dynamic tolling in the stochastic network environment. *Journal of Advanced Transportation*, 49(2):247–266, 2015.

[11] Li Yang, Romesh Saigal, and Hao Zhou. Distance-based dynamic pricing strategy for managed toll lanes. *Transportation Research Record: Journal of the Transportation Research Board*, (2283):90–99, 2012.

[12] Zhen Tan and H Oliver Gao. Hybrid model predictive control based dynamic pricing of managed lanes with multiple accesses. *Transportation Research Part B: Methodological*, 112:113–131, 2018.

[13] Mike Lindblom. I-405 toll lanes moving fast, but not as fast as Washington state law requires, 2017. URL https://www.seattletimes.com/seattle-news/transportation/i-405-toll-lanes-moving-fast-but-not-as-fast-as-state-law-requires/. Last Accessed: August 1, 2018.

[14] Luz Lazo. Virginia to tweak 66 express lanes pricing to address tolls that have topped $47. https://www.washingtonpost.com/local/trafficandcommuting/virginia-to-tweak-66-express-lanes-pricing-to-address-tolls-that-have-topped-47/2018/04/30/70441ab8-4c88-11e8-84a0-458a1aa9ac0a_story.html?noredirect=on&utm_term=.89f527d40d7d, 2018. Accessed January, 2020.

[15] Avinash Unnikrishnan and Steven Travis Waller. User equilibrium with recourse. *Networks and Spatial Economics*, 9(4):575, 2009.

[16] Tarun Rambha, Stephen D Boyles, Avinash Unnikrishnan, and Peter Stone. Marginal cost pricing for system optimal traffic assignment with recourse under supply-side uncertainty. *Transportation Research Part B: Methodological*, 110:104–121, 2018.

[17] Song Gao. Modeling strategic route choice and real-time information impacts in stochastic and time-dependent networks. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1298–1311, 2012.

[18] George H Polychronopoulos and John N Tsitsiklis. Stochastic shortest path problems with recourse. *Networks: An International Journal*, 27(2):133–143, 1996.

[19] S Travis Waller and Athanasios K Ziliaskopoulos. On the online shortest path problem with limited arc cost dependencies. *Networks: An International Journal*, 40(4):216–227, 2002.

[20] J Scott Provan. A polynomial-time algorithm to find shortest paths with recourse. *Networks: An International Journal*, 41(2):115–125, 2003.

[21] Venktesh Pandey and Stephen D Boyles. Dynamic pricing for managed lanes with multiple entrances and exits. *Transportation Research Part C: Emerging Technologies*, 96:304–320, 2018.

[22] Song Gao and Ismail Chabini. Optimal routing policy problems in stochastic time-dependent networks. *Transportation Research Part B: Methodological*, 40(2):93–122, 2006.

[23] Sathaporn Opasanon and Elise Miller-Hooks. Multicriteria adaptive paths in stochastic, time-varying networks. *European Journal of Operational Research*, 173(1):72–91, 2006.

[24] Elise D Miller-Hooks and Hani S Mahmassani. Least expected time paths in stochastic, time-varying transportation networks. *Transportation Science*, 34(2):198–215, 2000.

[25] Mark W Burris and John F Brady. Unrevealed preferences: Unexpected traveler response to pricing on managed lanes. *Transportation Research Record*, 2672(5):23–32, 2018.

[26] Lauren M Gardner, Hillel Bar-Gera, and Stephen D Boyles. Development and comparison of choice models and tolling schemes for high-occupancy/toll (HOT) facilities. *Transportation Research Part B: Methodological*, 55:142–153, 2013.

[27] Yingyan Lou, Yafeng Yin, and Jorge A Laval. Optimal dynamic pricing strategies for high-occupancy/toll lanes. *Transportation Research Part C: Emerging Technologies*, 19(1):64–74, 2011.

[28] Dimitra Michalaka, Yingyan Lou, and Yafeng Yin. Proactive and robust dynamic pricing strategies for high-occupancy-toll (HOT) lanes. In *Transportation Research Board 90th Annual Meeting*, number 11-2617, 2011.

[29] Daniele Pretolani. A directed hypergraph model for random time dependent shortest paths. *European Journal of Operational Research*, 123(2):315–324, 2000.

[30] A Arun Prakash. Pruning algorithm for the least expected travel time path on stochastic and time-dependent networks. *Transportation Research Part B: Methodological*, 108: 127–147, 2018.

[31] Elise Miller-Hooks. Adaptive least-expected time paths in stochastic, time-varying transportation and data networks. *Networks: An International Journal*, 37(1):35–52, 2001.

[32] Stephen D Boyles and S Travis Waller. Optimal information location for adaptive routing. *Networks and Spatial Economics*, 11(2):233–254, 2011.

[33] Ehsan Jafari and Stephen D Boyles. Online charging and routing of electric vehicles in stochastic time-varying networks. *Transportation Research Record: Journal of the Transportation Research Board*, (2667):61–70, 2017.

[34] A Arun Prakash and Karthik K Srinivasan. Finding the most reliable strategy on stochastic and time-dependent transportation networks: A hypergraph based formulation. *Networks and Spatial Economics*, 17(3):809–840, 2017.

[35] Kay Fitzpatrick, Marcus A Brewer, Susan Chrysler, Nick Wood, Beverly Kuhn, Ginger Goodin, Chuck Fuhs, David Ungemah, Benjamin Perez, Vickie Dewey, et al. *Guidelines for Implementing Managed Lanes*. Number Project 15-49. 2017.

[36] Song Gao and He Huang. Real-time traveler information for optimal adaptive routing in stochastic time-dependent networks. *Transportation Research Part C: Emerging Technologies*, 21(1):196–213, 2012.

[37] Danhong Cheng and Sherif Ishak. Maximizing toll revenue and level of service on managed lanes with a dynamic feedback-control toll pricing strategy. *Canadian Journal of Civil Engineering*, 43(1):18–27, 2015.

[38] Tomer Toledo, Omar Mansour, and Jack Haddad. Simulation-based optimization of HOT lane tolls. *Transportation Research Procedia*, 6:189–197, 2015.

[39] Stephen D Boyles, Lauren M Gardner, and Hillel Bar-Gera. Incorporating departure time choice into high-occupancy/toll (HOT) algorithm evaluation. *Transportation Research Procedia*, 9:90–105, 2015.

[40] Caner Göçmen, Robert Phillips, and Garrett van Ryzin. Revenue maximizing dynamic tolls for managed lanes: A simulation study. 2015.

[41] Lauren Gardner, Stephen D Boyles, Hillel Bar-Gera, and Kelly Tang. Robust tolling schemes for high-occupancy/toll (HOT) facilities under variable demand. *Transportation Research Record*, 2450:152–162, 2015.

[42] Dimitra Michalaka, Yafeng Yin, and David Hale. Simulating high-occupancy toll lane operations. *Transportation Research Record: Journal of the Transportation Research Board*, (2396):124–132, 2013.

[43] Elena G Dorogush and Alex A Kurzhanskiy. Modeling toll lanes and dynamic pricing control. *arXiv:1505.00506*, 2015.

[44] Apostolos Kotsialos, Markos Papageorgiou, Morgan Mangeas, and Habib Haj-Salem. Coordinated and integrated control of motorway networks via non-linear optimal control. *Transportation Research Part C: Emerging Technologies*, 10(1):65–84, 2002.

[45] M Papageorgiou and M Marinaki. A feasible direction algorithm for the numerical solution of optimal control problems. *Dynamic Syst. Simulation Lab., Tech. Univ. Crete, Chania, Greece*, 1995.

[46] Andreas Hegyi, Bart De Schutter, and Hans Hellendoorn. Model predictive control for optimal coordination of ramp metering and variable speed limits. *Transportation Research Part C: Emerging Technologies*, 13(3):185–209, 2005.

[47] Yihang Zhang and Petros A Ioannou. Combined variable speed limit and lane change control for truck-dominant highway segment. In *IEEE 18th International Conference on Intelligent Transportation Systems (ITSC), 2015*, pages 1163–1168. IEEE, 2015.

[48] Patrick Mannion, Jim Duggan, and Enda Howley. An experimental review of reinforcement learning algorithms for adaptive traffic signal control. In *Autonomic Road Transport Support Systems*, pages 47–66. Springer, 2016.

[49] Kasra Rezaee. *Decentralized coordinated optimal ramp metering using multi-agent reinforcement learning*. PhD thesis, University of Toronto (Canada), 2014.

[50] Francois Belletti, Daniel Haziza, Gabriel Gomes, and Alexandre M Bayen. Expert level control of ramp metering based on multi-task deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 19(4):1198–1207, 2017.

[51] Feng Zhu and Satish V Ukkusuri. Accounting for dynamic speed limit control in a stochastic traffic environment: A reinforcement learning approach. *Transportation research part C: emerging technologies*, 41:30–47, 2014.

[52] Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): methodology and large-scale application on downtown toronto. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1140–1150, 2013.

[53] Lior Kuyer, Shimon Whiteson, Bram Bakker, and Nikos Vlassis. Multiagent reinforcement learning for urban traffic control using coordination graphs. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 656–671. Springer, 2008.

[54] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 38(2):156–172, 2008.

[55] Jelle R Kok and Nikos Vlassis. Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research*, 7(Sep):1789–1828, 2006.

[56] Venktesh Pandey. Optimal Dynamic Pricing for Managed Lanes with Multiple Entrances and Exits. Master's thesis, The University of Texas at Austin, 2016.

[57] Carlos F Daganzo. The cell transmission model, part II: network traffic. *Transportation Research Part B: Methodological*, 29(2):79–93, 1995.

[58] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2011.

[59] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[60] Markos Papageorgiou, Habib Hadj-Salem, Jean-Marc Blosseville, et al. Alinea: A local feedback control law for on-ramp metering. *Transportation Research Record*, 1320(1): 58–67, 1991.

[61] John Miller. Dynamic pricing: Leveraging technology to manage pricing during changing conditions. In *Presented at 2017 IBTTA/TRB Joint Symposium on AET and Managed Lanes, Dallas, Texas*, page n.a. IBTTA, 2017.

[62] Carlos Guestrin, Michail Lagoudakis, and Ronald Parr. Coordinated reinforcement learning. In *ICML*, volume 2, pages 227–234, 2002.

[63] Venktesh Pandey and Stephen D Boyles. Multiagent reinforcement learning algorithm for distributed dynamic pricing of managed lanes. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2346–2351. IEEE, 2018.

[64] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[65] Soheil Mohamad Alizadeh Shabestary and Baher Abdulhai. Deep learning vs. discrete reinforcement learning for adaptive traffic signal control. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 286–293. IEEE, 2018.

[66] Cathy Wu, Aboudy Kreidieh, Kanaad Parvate, Eugene Vinitsky, and Alexandre M Bayen. Flow: Architecture and benchmarking for reinforcement learning in traffic control. *arXiv preprint arXiv:1710.05465*, 2017.

[67] Antonella Ferrara, Simona Sacone, and Silvia Siri. *Freeway traffic modelling and control*. Springer, 2018.

[68] Venktesh Pandey. Optimal dynamic pricing for managed lanes with multiple entrances and exits. Master's thesis, The University of Texas at Austin, 2016.

[69] Wade Genders and Saiedeh Razavi. Using a deep reinforcement learning agent for traffic signal control. *arXiv preprint arXiv:1611.01142*, 2016.

[70] Elise van der Pol. Deep reinforcement learning for coordination in traffic light control. *Master's thesis, University of Amsterdam*, 2016.

[71] Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[72] Kok-Lim Alvin Yau, Junaid Qadir, Hooi Ling Khoo, Mee Hong Ling, and Peter Komisarczuk. A survey on reinforcement learning models and algorithms for traffic signal control. *ACM Computing Surveys (CSUR)*, 50(3):34, 2017.

[73] Yundi Zhang, Bilge Atasoy, and Moshe Ben-Akiva. Calibration and optimization for adaptive toll pricing. In *2018 97th Annual Meeting of Transportation Research Board*, pages 18–05863. TRB, 2018.

[74] Venktesh Pandey and Stephen D. Boyles. Comparing route choice models for managed lane networks with multiple entrances and exits. *Transportation Research Record*, 2673 (10):381–393, 2019. doi: 10.1177/0361198119848706. URL https://doi.org/10.1177/0361198119848706.

[75] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[76] Elise Van der Pol and Frans A Oliehoek. Coordinated deep reinforcement learners for traffic light control. *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)*, 2016.

[77] John Schulman. *Optimizing expectations: From deep reinforcement learning to stochastic computation graphs*. PhD thesis, UC Berkeley, 2016.

[78] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[79] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[80] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[81] OpenAI. Welcome to Spinning Up in Deep RL– Spinning Up documentation. https://spinningup.openai.com/en/latest/index.html, 2019. Last Accessed: June 20, 2019.

[82] Chunming Liu, Xin Xu, and Dewen Hu. Multiobjective reinforcement learning: A comprehensive overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3):385–398, 2014.

[83] Robert B Dial. Bicriterion traffic assignment: basic theory and elementary algorithms. *Transportation science*, 30(2):93–111, 1996.

[84] Robert B Dial. Bicriterion traffic assignment: efficient algorithms plus examples. *Transportation Research Part B: Methodological*, 31(5):357–379, 1997.

[85] Hai Yang and Hai-Jun Huang. The multi-class, multi-criteria traffic network equilibrium and systems optimum problem. *Transportation Research Part B: Methodological*, 38(1):1–15, 2004.

[86] Anna Nagurney and June Dong. A multiclass, multicriteria traffic network equilibrium model with elastic demand. *Transportation Research Part B: Methodological*, 36(5): 445–469, 2002.

[87] Stella C Dafermos. Toll patterns for multiclass-user transportation networks. *Transportation science*, 7(3):211–223, 1973.

[88] Anna Nagurney. A multiclass, multicriteria traffic network equilibrium model. *Mathematical and Computer Modelling*, 32(3-4):393–411, 2000.

[89] Younes Hamdouch and Siriphong Lawphongpanich. Congestion pricing for schedule-based transit networks. *Transportation Science*, 44(3):350–366, 2010.

[90] Yosef Sheffi. *Urban transportation networks*. Prentice-Hall, Englewood Cliffs, NJ, 1985.

[91] Stephen D. Boyles. *Operational, supply-side uncertainty in transportation networks: causes, effects, and mitigation strategies*. PhD thesis, The University of Texas at Austin, 2009.

[92] Hillel Bar-Gera. Traffic assignment by paired alternative segments. *Transportation Research Part B: Methodological*, 44(8-9):1022–1046, 2010.

[93] Robert B Dial. A path-based user-equilibrium traffic assignment algorithm that obviates path storage and enumeration. *Transportation Research Part B: Methodological*, 40(10):917–936, 2006.

[94] Tarun Rambha. *Dynamic congestion pricing in within-day and day-to-day network equilibrium models*. PhD thesis, The University of Texas at Austin, 2016.

[95] Carlos F Daganzo and Yosef Sheffi. On stochastic models of traffic assignment. *Transportation science*, 11(3):253–274, 1977.

[96] Stein W Wallace. Decision making under uncertainty: Is sensitivity analysis of any use? *Operations Research*, 48(1):20–25, 2000.

[97] David Watling and Martin L Hazelton. The dynamics and equilibria of day-to-day assignment models. *Networks and Spatial Economics*, 3(3):349–370, 2003.

[98] Gary A Davis and Nancy L Nihan. Large population approximations of a general stochastic traffic assignment model. *Operations Research*, 41(1):169–178, 1993.

[99] Martin L Hazelton and David P Watling. Computation of equilibrium distributions of markov traffic-assignment models. *Transportation Science*, 38(3):331–342, 2004.

[100] Avinash Unnikrishnan and Dung-Ying Lin. User equilibrium with recourse: continuous network design problem. *Computer-Aided Civil and Infrastructure Engineering*, 27(7): 512–524, 2012.

[101] Dan Calderone and S Shankar Sastry. Markov decision process routing games. In *Proceedings of the 8th International Conference on Cyber-Physical Systems*, pages 273–279. ACM, 2017.

[102] Roger L Tobin and Terry L Friesz. Sensitivity analysis for equilibrium network flow. *Transportation Science*, 22(4):242–250, 1988.

[103] Michael Patriksson. Sensitivity analysis of traffic equilibria. *Transportation Science*, 38(3):258–281, 2004.

[104] Magnus Josefsson and Michael Patriksson. Sensitivity analysis of separable traffic equilibrium equilibria with application to bilevel optimization in network design. *Transportation Research Part B: Methodological*, 41(1):4–31, 2007.

[105] Ehsan Jafari and Stephen D Boyles. Improved bush-based methods for network contraction. *Transportation Research Part B: Methodological*, 83:298–313, 2016.

[106] Jian Wang, Srinivas Peeta, and Xiaozheng He. Multiclass traffic assignment model for mixed traffic flow of human-driven vehicles and connected and autonomous vehicles. *Transportation Research Part B: Methodological*, 126:139–168, 2019.

[107] Sarah HQ Li, Daniel Calderone, Lillian Ratliff, and Behcet Acikmese. Sensitivity analysis for Markov decision process congestion games. *arXiv preprint arXiv:1909.04167*, 2019.

[108] DW Hearn. Practical and theoretical aspects of aggregation problems in transportation planning models. *Publication of: Elsevier Science Publishers BV*, 1984.

[109] Stephen D Boyles. Bush-based sensitivity analysis for approximating subnetwork diversion. *Transportation Research Part B: Methodological*, 46(1):139–155, 2012.

[110] Ehsan Jafari, Venktesh Pandey, and Stephen D Boyles. A decomposition approach to the static traffic assignment problem. *Transportation Research Part B: Methodological*, 105:270–296, 2017.

[111] Martin L Puterman. *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[112] R Jayakrishnan, Wei T Tsai, Joseph N Prashker, and Subodh Rajadhyaksha. A faster path-based algorithm for traffic assignment. 1994.

[113] Stephen D. Boyles, Nicholas E. Lownes, and A. Unnikrishnan. *Transportation Network Analysis*, volume 1. 0.85 edition, 2020.

[114] Shu Lu and Yu Marco Nie. Stability of user-equilibrium route flow solutions for the traffic assignment problem. *Transportation Research Part B: Methodological*, 44(4): 609–617, 2010.

[115] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 22–31. JMLR. org, 2017.

[116] Kara M Kockelman and Jason D Lemp. Anticipating new-highway impacts: Opportunities for welfare analysis and credit-based congestion pricing. *Transportation Research Part A: Policy and Practice*, 45(8):825–838, 2011.