Enabling Time-Dependent Uncertain Edge Weights and Stochastic Routing in Road Networks

Yu Ma

PhD Dissertation

Department of Computer Science
Aarhus University
Denmark
Enabling Time-Dependent Uncertain Edge Weights and Stochastic Routing in Road Networks

A Dissertation
Presented to the Faculty of Science and Technology of Aarhus University in Partial Fulfillment of the Requirements for the PhD Degree

by
Yu Ma
July 27, 2015
Abstract

Data that describes the driving of vehicles in road networks, notably GPS data, is becoming increasingly available. Large volumes of such data allow us to better capture and understand the dynamic and uncertain traffic patterns that occur in road networks. Based on the availability of large volumes of GPS data, the dissertation proposes techniques that enable the efficient and accurate modeling of vehicular travel costs associated with the traversal of edges in road networks as time-dependent random variables, instead of as static single values. The key travel costs considered are travel time and environmental impact (fuel consumption or greenhouse gas emissions, also called eco-weights). We study how to support time-dependent stochastic routing in road networks where edges are associated with time-dependent uncertain travel costs, and we also provide a foundation for personalized and context-aware routing in this setting.

First, we propose a framework, called EcoMark, for the evaluation of so-called vehicular environmental impact models that aim to quantify the greenhouse gas emissions of a vehicle based on GPS data from the vehicle and a 3D model of the underlying road network. We apply EcoMark to eleven existing impact models to investigate their capabilities and performance and to gain insight into the effectiveness of EcoMark.

Second, we study how to use historical GPS data in order to assign time-dependent, uncertain eco-weights, which are sequences of histograms, to the edges in a road network. Different compression techniques are used to achieve compact histograms while retaining their accuracy. In addition, so-called virtual edges and extended virtual edges are proposed to represent adjacent edges with dependent travel costs.

Third, assuming a road network with time-varying, uncertain edge weights, we define a time-dependent, non-dominated stochastic routing problem. We then present an efficient method based on time-dependent uncertain contraction hierarchies that is able to find all paths between a source-destination pair at a given start time whose travel costs are not stochastically dominated by any other path.

Finally, again in a setting where time-dependent and uncertain travel costs, such as travel time and environmental impact, are captured, we provide techniques capable of capturing the behaviors of different drivers in terms of the travel costs. Further, we propose techniques that are able to identify a driver’s contexts and associated driving preferences using historical trajectories from the driver. These techniques are foundations for personalized and context-aware routing.
**Resumé**

Data, der beskriver køretøjers kørsel i vejnetværk, specielt GPS-data, bliver tilgængelige i større og større omfang. Store mængder af sådanne data give os mulighed for bedre at beskrive og forstå dynamiske og usikre trafikmønstre i vejnet. Baseret på tilgængeligheden af omfattende mængder af GPS-data, bidrager afhandlingen med teknikker, der muliggør effektivt og præcist modellering af omkostninger forbundet med kørsel på vejsegmenter som tidsafhængige stokastiske variable, i stedet for som statiske enkelte værdier. De vigtigste omkostninger i afhandlingen er rejsetid og miljøpåvirkning (brændstofforbrug eller udledning af drivhugasser, også kaldet økovægte). Vi beskriver hvordan man muliggør tidsafhængig stokastisk navigation i vejnet, hvor tidsafhængige usikre rejseomkostninger er knyttet til vejsegmenterne, og vi beskriver også et fundament for personlig og kontekstafhængig navigation i denne sammenhæng.

Først foreslår vi en løsning, kaldet EcoMark, der muliggør evaluering af miljøpåvirkningsmodeller, der har til formål at kvantificere et køretøjers udledninger af drivhugasser baseret på GPS-data fra køretøjet og en 3D-model af det underliggende vejnet. Vi anvender EcoMark på elleve eksisterende miljøpåvirkningsmodeller for at undersøge deres egenskaber og ydeevne og for at få indsigt i EcoMarks egenkaber.

Dernæst studerer vi hvordan man bruger historiske GPS-data med henblik på at knytte tidsafhængige, usikre økovægte, som er sekvenser af histogrammer, til segmenterne i et vejnet. Forskellige kompressionsteknikker anvendes til at opnå kompakte histogrammer samtidig med at deres nøjagtighed bevaes. Desuden foreslås såkaldte virtuelle vejsegmenter og udvidede virtuelle vejsegmenter der har til formål at repræsentere vejsegmenter med afhængige økovægte.

For det tredje definerer vi problemet at udføre tidsafhængig, ikke-domineret stokastisk navigation i vejnet med tidsvarierende, usikre vægtte. Derefter præsenterer vi en effektiv metode baseret på såkaldte tidsafhængige usikre "contraction hierarchies", der er i stand til at finde alle veje, der forbinder et startsted og en destination på et givet starttidspunkt og som har rejseomkostninger, der ikke er stokastisk domineret af en anden vejs rejseomkostninger.

Endelig bidrager vi med teknikker, der er i stand til at beskrive forskellige bilisters adfærd i form af rejseomkostninger. Det gøres igen i en sammenhæng, hvor tidsafhængig og usikre rejseomkostninger såsom rejsetid og miljøpåvirkning er tilgængelige. Endvidere foreslår vi teknikker, der er i stand til at identificere en bilists kontekster og tilhørende kørselspræferencer ved hjælp af historiske GPS-data fra bilis-
ten. Disse teknikker udgør en del af grundlaget for personlig og kontekstafhængig navigation.
Acknowledgments

I would like to thank the people who helped and supported me during my PhD studies, my PhD career would not be complete without your help and support.

First and foremost, I want to gratefully thank my advisor Christian S. Jensen for providing me with advices, comments, and guidance. He has always been supportive when we developed and discussed the research ideas. He also helped me to make my manuscripts intact and easy to read. His high standards for research have driven me to the next level not only as a PhD student, but also as a person.

Next, I would like to thank every colleague in Reduction project at Aarhus University, including Bin Yang, Manohar Kaul, and Chenjuan Guo. I want to give special thanks to Bin Yang, not only I learned very valuable writing skills from Bin, but I also developed my research ideas throughout discussions with him. Bin also invested a lot of time to improve all my manuscripts before Christian made the changes to them.

I am thankful to my Ph.D. support group, consisting of Larse Arge and Erik Ernst, who gave me useful comments and guidance on tracking my progress.

I also want to thank all my colleagues at the Data Intensive Systems Group, who are Manohar Kaul, Qiang Qu, Anders Skovsgaard, Laura Radaelli, Vaida Ceikute, Bin Yang, Chenjuan Guo, Darius Sidlauskas, Barbora Micenkova, Michael Lind Mortensen, Peiman Barnaghi, Matteo Magnani, Xuan-Hong Dung, Sean Chester, Kenneth Bogh, Leon Derczynski, Jan Neerbek, and Mai Thai Son. And especially thank Kenneth Bogh for fixing all hardware problems of the development server when I was not in Aarhus. Also I appreciate the kind help and discussions from my colleagues during my stay at Aalborg University, including Benjamin Krogh, Jilin Hu, Tanvir Ahmed, and Asif Baba.

Last but not least, I thank my family for their eternal love and support. In particular, I would like to thank my beloved wife, Yining (Katherine) Ma, for all her understanding and sacrifices, she quit her job in China to join me in Denmark, and we also moved to USA with me for an internship during the my PhD studies. Her encourage and love have always been my driving force in the past years, and thanks for always being caring and staying with me.

Thank you all.

Yu Ma,
# Contents

<table>
<thead>
<tr>
<th>Abstract i</th>
<th>Resumé iii</th>
<th>Acknowledgments v</th>
<th>Contents vii</th>
</tr>
</thead>
</table>

## I Overview

### 1 Introduction

1.1 EcoMark

1.1.1 Background and Motivation

1.1.2 Our Proposed Solution

1.1.3 Key Findings from Empirical Study

1.1.4 Future Work

1.2 Time-Dependent Uncertain Eco Weights

1.2.1 Background and Motivation

1.2.2 Our Proposed Solution

1.2.3 Key Findings from Empirical Study

1.2.4 Future Work

1.3 Time-Dependent Stochastic Routing

1.3.1 Background and Motivation

1.3.2 Our Proposed Solution

1.3.3 Key Findings from Empirical Study

1.3.4 Future Work

1.4 Context-aware, Personalized Routing

1.4.1 Background and Motivation

1.4.2 Our Proposed Solution

1.4.3 Key Findings from Empirical Study

1.4.4 Future Work

1.5 Dissertation Organization
# II Publications

## 2 Ecomark: Evaluating Models of Vehicular Environmental Impact  

### 2.1 Introduction  

### 2.2 Related Work  

### 2.3 EcoMark Design  

#### 2.3.1 EcoMark Overview  

#### 2.3.2 Trajectories  

#### 2.3.3 Modeling a 3D Spatial Network  

#### 2.3.4 Realizing a 3D Spatial Network  

### 2.4 Model Analysis  

#### 2.4.1 Instantaneous Models  

##### 2.4.1.1 EMIT  

##### 2.4.1.2 VT-Micro  

##### 2.4.1.3 MEF  

##### 2.4.1.4 SP  

##### 2.4.1.5 Joumard  

##### 2.4.1.6 SIDRA-Inst  

#### 2.4.2 Aggregated Models  

##### 2.4.2.1 Song  

##### 2.4.2.2 Tavares  

##### 2.4.2.3 SIDRA-4Mode  

##### 2.4.2.4 SIDRA-Running  

##### 2.4.2.5 SIDRA-Avg  

### 2.4.3 Summary  

### 2.5 Empirical Studies  

#### 2.5.1 Setup  

#### 2.5.2 Evaluating Instantaneous Models  

#### 2.5.3 Evaluating Aggregated Models  

#### 2.5.4 Aggregation of Instantaneous Models Versus Aggregated Models  

#### 2.5.5 Effect of Road Grades  

#### 2.5.6 Empirical Findings  

### 2.6 Conclusions  

## 3 Enabling Time-Dependent Uncertain Eco-Weights For Road Networks  

### 3.1 Introduction  

### 3.2 Related Work  

### 3.3 Problem Setting and Definition  

#### 3.3.1 Time-Dependent Histograms  

#### 3.3.2 Road Networks and Trajectories  

#### 3.3.3 Problem Definition and Solution Framework  

### 3.4 ERN Construction  

#### 3.4.1 Initial Time Dependent Histograms  

---

*CONTENTS*

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>II</td>
<td>Publications</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>Ecomark: Evaluating Models of Vehicular Environmental Impact</td>
<td>21</td>
</tr>
<tr>
<td>2.1</td>
<td>Introduction</td>
<td>21</td>
</tr>
<tr>
<td>2.2</td>
<td>Related Work</td>
<td>23</td>
</tr>
<tr>
<td>2.3</td>
<td>EcoMark Design</td>
<td>24</td>
</tr>
<tr>
<td>2.3.1</td>
<td>EcoMark Overview</td>
<td>24</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Trajectories</td>
<td>26</td>
</tr>
<tr>
<td>2.3.3</td>
<td>Modeling a 3D Spatial Network</td>
<td>26</td>
</tr>
<tr>
<td>2.3.4</td>
<td>Realizing a 3D Spatial Network</td>
<td>27</td>
</tr>
<tr>
<td>2.4</td>
<td>Model Analysis</td>
<td>28</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Instantaneous Models</td>
<td>28</td>
</tr>
<tr>
<td>2.4.1.1</td>
<td>EMIT</td>
<td>28</td>
</tr>
<tr>
<td>2.4.1.2</td>
<td>VT-Micro</td>
<td>29</td>
</tr>
<tr>
<td>2.4.1.3</td>
<td>MEF</td>
<td>29</td>
</tr>
<tr>
<td>2.4.1.4</td>
<td>SP</td>
<td>30</td>
</tr>
<tr>
<td>2.4.1.5</td>
<td>Joumard</td>
<td>30</td>
</tr>
<tr>
<td>2.4.1.6</td>
<td>SIDRA-Inst</td>
<td>30</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Aggregated Models</td>
<td>31</td>
</tr>
<tr>
<td>2.4.2.1</td>
<td>Song</td>
<td>31</td>
</tr>
<tr>
<td>2.4.2.2</td>
<td>Tavares</td>
<td>31</td>
</tr>
<tr>
<td>2.4.2.3</td>
<td>SIDRA-4Mode</td>
<td>31</td>
</tr>
<tr>
<td>2.4.2.4</td>
<td>SIDRA-Running</td>
<td>32</td>
</tr>
<tr>
<td>2.4.2.5</td>
<td>SIDRA-Avg</td>
<td>33</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Summary</td>
<td>33</td>
</tr>
<tr>
<td>2.5</td>
<td>Empirical Studies</td>
<td>34</td>
</tr>
<tr>
<td>2.5.1</td>
<td>Setup</td>
<td>34</td>
</tr>
<tr>
<td>2.5.2</td>
<td>Evaluating Instantaneous Models</td>
<td>35</td>
</tr>
<tr>
<td>2.5.3</td>
<td>Evaluating Aggregated Models</td>
<td>37</td>
</tr>
<tr>
<td>2.5.4</td>
<td>Aggregation of Instantaneous Models Versus Aggregated Models</td>
<td>41</td>
</tr>
<tr>
<td>2.5.5</td>
<td>Effect of Road Grades</td>
<td>42</td>
</tr>
<tr>
<td>2.5.6</td>
<td>Empirical Findings</td>
<td>42</td>
</tr>
<tr>
<td>2.6</td>
<td>Conclusions</td>
<td>43</td>
</tr>
<tr>
<td>3</td>
<td>Enabling Time-Dependent Uncertain Eco-Weights For Road Networks</td>
<td>45</td>
</tr>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>45</td>
</tr>
<tr>
<td>3.2</td>
<td>Related Work</td>
<td>47</td>
</tr>
<tr>
<td>3.3</td>
<td>Problem Setting and Definition</td>
<td>48</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Time-Dependent Histograms</td>
<td>49</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Road Networks and Trajectories</td>
<td>49</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Problem Definition and Solution Framework</td>
<td>50</td>
</tr>
<tr>
<td>3.4</td>
<td>ERN Construction</td>
<td>51</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Initial Time Dependent Histograms</td>
<td>51</td>
</tr>
</tbody>
</table>
3.4.2  Histogram Merging .......................................................... 51
3.4.3  Bucket Reduction ........................................................... 53
3.5  GHG Emissions Estimation .................................................. 55
  3.5.1  Modeling Dependence Among Adjacent Edges ...................... 55
    3.5.1.1  Dependence Analysis ........................................... 55
    3.5.1.2  Virtual Edges .................................................. 56
    3.5.1.3  Extended Virtual Edges ....................................... 57
    3.5.1.4  Sub-Routes .................................................... 60
  3.5.2  Histogram Aggregation ................................................ 61
  3.5.3  GHG Emissions Estimation for Routes ............................... 63
3.6  Empirical Study ............................................................... 66
  3.6.1  Experimental Settings ............................................... 67
  3.6.2  Running Time Efficiency ............................................ 68
  3.6.3  Estimation Accuracy ............................................... 69
  3.6.4  Storage Consumption .............................................. 73
  3.6.5  Summary .............................................................. 74
3.7  Conclusions ................................................................. 75

4  A Practical Approach to Routing With Time-Varying, Uncertain Edge
Weights ................................................................. 77
  4.1  Introduction ............................................................. 77
4.2  Preliminaries ............................................................... 79
  4.2.1  Road Networks, Paths, and Trajectories .......................... 79
  4.2.2  Time-dependent Uncertain Edge Weights .......................... 81
    4.2.2.1  Uncertain Edge Weights Representation ..................... 81
  4.2.3  Uncertain Edge Weight Convolution ................................ 83
  4.2.4  Problem Definition ................................................ 85
  4.2.5  Framework Overview .............................................. 86
4.3  Query Processing ........................................................... 86
  4.3.1  Baseline Method .................................................... 86
  4.3.2  Contraction Hierarchies .......................................... 90
  4.3.3  Time-dependent Uncertain Contraction Hierarchy ............... 94
  4.3.4  CH-Based Method .................................................. 98
  4.3.5  Speedup Techniques .............................................. 101
4.4  Empirical Study ............................................................ 101
  4.4.1  Experimental Settings ............................................ 102
  4.4.2  Concrete Query Example .......................................... 103
  4.4.3  Query Performance ............................................... 104
    4.4.3.1  Distance Between Sources and Destinations .......... 104
    4.4.3.2  Peak vs. Off-Peak Hours .................................. 105
    4.4.3.3  Varying the Histogram Bucket Count .................... 106
    4.4.3.4  Varying the Uncertain Weight Percentage ............... 106
  4.4.4  TNSP Count ....................................................... 106
    4.4.4.1  Distance Between Sources and Destinations ......... 106
Part I

Overview
Chapter 1

Introduction

In the past few years, traffic data from large road networks has become increasingly available due to the expanding use of sensors, scanners, and location-aware communication devices, i.e., GPS devices attached to vehicles, and it is now possible to collect tremendous amounts of high-frequency GPS data from vehicles when they travel in road networks at very low cost.

The availability of a large amounts of GPS data has provided the opportunity for much deeper insight into the traffic in road networks. First, we can not only quantify the travel time cost of traversing an edge in a road network but also compute the environmental travel costs using appropriate environmental travel cost evaluation models. The modeling of environmental travel costs is fundamental to enable eco-routing and can potentially contribute to reducing greenhouse gas (GHG) emissions from vehicles. Second, instead of using static single values as travel costs, massive GPS data make it possible to model travel costs in more details. For instance, the time dependence and uncertainty of the travel costs can now be taken into account and we can assign time-varying random variables as the travel costs to the edges. Third, based on these time-varying uncertain travel costs, we can design techniques to solve routing problems in settings that simulate real traffic scenarios.

Many studies have been conducted regarding large scale traffic data sets. For example, to measure the vehicular environmental travel impact, various models [15] have been proposed to quantify fuel consumption and GHG emissions of based on GPS data from vehicles. These models consider a wide range of factors, such as vehicle speed and acceleration, different physical features of vehicles, personal driving behavior of the drivers, and geometric information of the road network. However, while there are many different environmental travel impact models exist, there lacks a comprehensive evaluation and comparison of these models.

In addition, a number of studies involve time-dependent and uncertain travel costs of edges in road networks, but due to the lack of real traffic data, most previous efforts rely on synthetically generated travel costs. However, using historical GPS data, a recent study [91] assigns time-dependent, uncertain travel costs to connections between landmarks and estimates the travel cost distributions between landmarks at
CHAPTER 1. INTRODUCTION

a given time. The study simply assumes that the travel costs are independent. In addition, as the study assign weights to each landmark connection rather than each edge in the road network, it is not applicable for routing.

Path planning has been a popular research topic for decades. Newly invented indexing and pre-processing techniques [13] can solve shortest path problem for continental road networks in microseconds, when the edge weights are modeled as static single values. Furthermore, a few studies propose routing techniques for road networks with time-dependent and uncertain edge weights [27, 43, 64], but there is no existing study that relies on large road networks with time-dependent uncertain edge weights generated from real traffic data.

Motivated by the above observations, we identified the need to further utilize traffic data to better model travel costs in road networks and to design new routing techniques for the resulting models. Specifically, the goals of this thesis include:

- Analyze and evaluate the existing vehicular environmental impact models.
- Assign time-dependent uncertain edge weights to reflect the dynamics of traffic in road networks.
- Invent stochastic routing algorithms for large road networks with time-dependent uncertain edge weights.
- Identify appropriate contexts and apply preferences in the contexts for individual drivers, thus enabling personalized and context-aware routing in road networks with time-dependent uncertain edge weights.

The remainder of the chapter is organized as follows. Section 1.1 describes our framework, called EcoMark, for vehicular environmental impact model evaluation. Section 1.2 describes the creation and use of time-dependent uncertain edge weights in road networks. Section 1.3 describes the path-finding in large road network while considering the time dependence and uncertainty of travel costs. Section 1.4 describes how we identify context-aware driving preferences for individual drivers from historical GPS trajectories, and it proposes techniques to identify a driver’s contexts and to identify driving preferences for each context using historical trajectories from the driver.

1.1 EcoMark

1.1.1 Background and Motivation

Reduction in the greenhouse gas (GHG) emissions is crucial for combating global warming that has increasingly adverse effects on life on Earth. The transportation sector is the second largest in terms of GHG emissions. Eco-routing and eco-driving are simple yet effective approaches to reduce fuel consumption and GHG emissions
from road transportation. Eco-driving targets eco-friendly driver behavior, and eco-routing recommends routes that aim to minimize fuel consumption and GHG emissions. In particular, the fundamental step towards eco-routing is to compute the environmental footprints of the vehicles in a road network. Although a range of vehicular environmental impact models are available, we lack a comprehensive analysis on these models. GPS trajectory data from vehicles moving in a road network makes it possible to systematically measure the vehicular environmental travel costs and consequently compare the existing vehicular environmental impact models.

1.1.2 Our Proposed Solution

EcoMark is designed to evaluate state-of-the-art models of vehicular environmental impact in terms of fuel consumption and GHG emissions. It provides an understanding of the utility of the impact models in relation to eco-driving and eco-routing and offers insight into aspects such as which models can be used for identifying relationships between environmental impact and driver behavior, thus enabling eco-driving, and which models are suitable for assigning weights to road segments that capture environmental impact, thus enabling eco-routing.

Figure 1.1 depicts an overview of EcoMark. Three raw data sets are used: a set of GPS observations, a 2D spatial network, and a laser scan point cloud. A map matching module takes as input the set of GPS observations and the 2D spatial network. It outputs a set of map matched trajectories. A 3D spatial network generation module creates a 3D spatial network from the 2D spatial network and the laser scan point cloud. The trajectories and the 3D spatial network are fed into EcoMark as input data.

The eleven models considered in EcoMark take as input traffic and road information that can be obtained from GPS trajectories and a 3D spatial network. These models are categorized into instantaneous models and aggregated models. The instantaneous models take as input instantaneous (i.e., second-by-second) velocities and accelerations and output instantaneous fuel usage or GHG emissions. In contrast, the aggregated models take as input average velocities and output aggregated fuel usage or GHG emissions. The aggregated models can be applied at different aggregation levels, e.g., at the level of road segments or at the level of longer routes.

EcoMark is used to perform the following comparisons and analyses: (1) comparison and analysis of instantaneous models; (2) comparison and analysis of aggregated models; (3) aggregation of the instantaneous results, and comparison of them with the results obtained from the aggregated models; (4) comparison of the (instantaneous and aggregated) results with and without the use of road grades.

Contributions:

- We propose a sophisticated evaluation framework that encompasses a 3D road network model.
• We provide a categorization and comparison of all eleven known models that can estimate fuel usage and GHG emissions is conducted.

• We conduct comprehensive experimental studies on a half-year collection of GPS data from vehicles traveling in North Jutland, Denmark.

1.1.3 Key Findings from Empirical Study

We use GPS data collected from 150 vehicles traveling in North Jutland, Denmark, during January to June 2007. The GPS data covers a variety of traffic conditions, e.g., peak and off-peak hour traffic, highway traffic, and arterial road traffic. The sampling frequency is 1 Hz, which makes application to the instantaneous models easy. We apply an existing map matching tool [67] along with a 2D spatial network of North Jutland, Denmark obtained from OpenStreetMap to the GPS data, from which we get a set of 52,084 trajectories. The main findings are as follows:

• Instantaneous velocities and accelerations reflect driving behaviors, so instantaneous models can be used to measure the environmental impact of different such behaviors. Along with classical data mining methods, instantaneous models can be used to classify good and bad driving behaviors in terms of en-
1.2. TIME-DEPENDENT UNCERTAIN ECO WEIGHTS

environmental impact. Thus, we can suggest good driving behaviors to drivers and the instantaneous models are appropriate for eco-driving applications.

- Aggregated models estimate environmental impact per unit length. This type of impact can be used to assigning eco-weights to road segments to enable eco-routing. Moreover, instantaneous impact on a road segment predicted by an instantaneous model, can be aggregated and then used as the eco-weight of the road segment. We also suggest that a single eco-weight per road segment suffices in some cases, while time-dependent weights or a range of weights should be considered in other cases.

- Road grades substantially affect the environmental impact predicted by both instantaneous and aggregated models. Therefore, the use of 3D spatial networks benefits both eco-driving and eco-routing.

1.1.4 Future Work

It is relevant to study how well the models predict actual environmental impact. Such a study is possible if vehicular CAN bus data that records the actual GHG emissions and fuel usage along with corresponding GPS observations is available.

1.2 Time-Dependent Uncertain Eco Weights

1.2.1 Background and Motivation

Given a source-destination pair in a road network, there are different routes between the source and destination, such as the fastest route, shortest route, and eco route. The eco route is the most environmentally friendly route, i.e., the route that produces the least GHG emissions [16]. Neither the shortest nor the fastest routes generally have the least environmental impact [16]. Thus, eco route is not always the same as the fastest or shortest route for a source-destination pair. Figure 1.2 shows an example of the shortest route, the fastest route, and the eco-route between source A and destination D.

Based on the findings from Section 1.1, we can identify the most appropriate evaluation model to measure the enviromental travel costs for all the edges in our Denmark road network with a large GPS data set. However, due the dynamics of real world traffic, a single valued edge weight does not fully reflect the traffic conditions. On the one hand, the travel cost for an edge may vary between peak hours and off-peak hours. On the other hand, an edge weight may also be uncertain as different drivers have different driving behaviors. Thus, time-dependent uncertain weights is an alternative way to model edge weights in a road network, and it is a key step to enable eco-routing. To the best of our knowledge, there is only one closely related study, T-drive [91], which aims at providing a fastest-route service based on travel-time weights learned from GPS records obtained from taxis. Rather than assign costs to each road-network edge, T-drive identifies so-called landmarks and assigns costs
CHAPTER 1. INTRODUCTION

Figure 1.2: Eco-Route, Fastest Route, and Shortest Route

to travel between pairs of such landmarks. Specifically, is assigns several histograms to a connection between two landmarks, where each histogram represents the distribution of travel times during a time interval. In doing so, T-drive uses the same buckets for the histograms on a landmark connection during different intervals.

1.2.2 Our Proposed Solution

A GPS record \( r_i \) specifies the location (typically with latitude and longitude coordinates) and velocity of a vehicle at a particular time \( r_i.t \). A Trajectory \( tr = (r_1, r_2, \ldots, r_x) \) consists of a sequence of GPS records. Furthermore, the GPS records in a trajectory are ordered based on their timestamps. A GPS record in a trajectory can be mapped to a specific location on an edge in the road network using a map-matching algorithm [67].

Given a set of map matched trajectories \( TR \) in a road network \( G' = (V, E, null) \), where \( V \) and \( E \) are the vertex and edge sets in \( G' \), we study how to obtain the corresponding Eco-Road Network \( ERN G = (V, E, F) \). Specifically, the key task is to determine \( G.F \), which assigns time dependent histograms to edges, based on trajectory set \( TR \). First, we transform the map-matched trajectories into a set \( TRR \) of traversal records of the form \( trr = (e, t_s, tt, ge, trj_j) \). A traversal record \( r \) indicates that edge \( e \) is traversed by trajectory \( trj_j \) starting at time \( t_s \). The travel time and the GHG emissions of the traversal are \( tt \) and \( ge \), respectively. Then, we use a sequence of time-dependent histograms \( \langle H_1, H_2, \ldots, H_{N-1}, H_N \rangle \) to represent each edge’s time-dependent uncertain travel cost. In doing so, we first partition time into \( N \) contiguous intervals of interest, and for each edge \( e \), we assign a histogram \( H \) for a specific time interval as the travel cost to traverse that edge during the time interval.
A histogram $H$ is built using all traversal records that occurred on the edge during the time interval of interest.

After we get initial histograms for each edge in the road network, different compression techniques are proposed to gain a compact yet accurate representation of the road network. In particular, if temporally adjacent histograms $H_i$ and $H_{i+1}$ on an edge represent similar data distributions for time periods $T_i$ and $T_{i+1}$, histogram merging is used to merge the two histograms into one histogram $\overline{H}$ that represents the data distribution for the longer period $T = T_i \cup T_{i+1}$. Plus, bucket reduction transforms a histogram $H$ into a new histogram $\hat{H}$ that approximates the data distribution represented by $H$ using fewer buckets.

We also analyze and measure the travel cost dependence on adjacent edges using Normalized Mutual Information of travel cost distributions of two adjacent edges. Virtual edges and extended virtual edges are proposed to represent adjacent edges with highly dependent travel costs. Figure 1.3 gives an overview of how to obtain an Eco-Road Network (ERN).

With the above techniques, we propose methods to estimate the GHG emissions for a route with multiple edges at a given time, where the estimated travel costs are also represented as histograms and the aggregation of two adjacent edges’ travel costs are represented as the convolution of their corresponding histograms.
Contributions:

- We propose an Eco Road Network as a foundation for enabling eco-routing.

- Compact, time-dependent histograms are proposed to represent the time-dependent, uncertain eco-weights of edges.

- By introducing virtual edges and extended virtual edges, we make it possible to capture the dependencies among the eco-weights of adjacent edges.

- We propose several histogram aggregation methods that are able to estimate GHG emissions of routes based on the eco-weights of edges, virtual edges, and extended virtual edges.

- Experiments are conducted on a comprehensive GPS data set that provide insight into the efficiency and accuracy of the paper’s proposals.

1.2.3 Key Findings from Empirical Study

We use a large GPS tracking data set containing more than 200 million GPS records collected at 1 Hz from 150 vehicles in Denmark from January 2007 to December 2008. A total of 802K traversal records are generated from the data set. We use the road network of Denmark from OpenStreetMap\[1\] To get the best map-matching, we extract edges from OpenStreetMap data with the finest granularity, with 414K vertices and 1,628K edges. Moreover, based on our experimental results, we propose recommended parameter settings to achieve a compact representation of an ERN while retaining high GHG emissions accuracy for routes. The main findings are as follows:

- The average time to generate GHG emissions histograms for a single edge is 72 microseconds, and the average histogram merging time is 9 microseconds.

- We define histogram approximation accuracy as the distance between the original data distribution and the derived histogram representations, and our study shows that we can achieve good estimation accuracy for routes by modeling adjacent edges with high dependency as virtual edges and extended virtual edges.

- With our recommended settings, each edge requires on average 3.98 histograms and about 0.61 KB storage space.

\[1\] http://www.openstreetmap.org/
1.2.4 Future Work

It is of interest to explore advanced routing algorithms that can more utilize the time-dependent, uncertain eco-weights, e.g., to compute stochastic eco-routes. Additionally, using an inverted approach that assigns a time-dependent histogram based eco-weight to a group of edges that share similar travel cost distributions may result in further storage space reductions.

1.3 Time-Dependent Stochastic Routing

1.3.1 Background and Motivation

Path planning is an indispensable part of many location-based services and is being used increasingly as Internet-enabled mobile devices continue to proliferate. Although useful route planning methods do exist, to fulfill user needs, we still need to address the following two challenges.

**Time-Dependent Uncertain Travel Costs:** Path planning relies on a weighted graph representation of a road network, where the weight of an edge (i.e., a road segment) refers to the travel cost of traversing the edge. Most existing work uses *deterministic weight modeling*, where the travel costs of traversing edges (i.e., edge weights) are modeled as deterministic values. For instance, several major on-line service providers (e.g., Google Maps, Bing Maps) employ the lengths of road segments divided by the speed limits of the road segments as the travel time based edge weights [92]. However, deterministic weight modeling does not accurately capture the traffic conditions in road networks. On the one hand, the cost to traverse an edge may vary during a day, i.e., the average travel time during peak hours is higher than that of off-peak hours. On the other hand, while traversing the same edge, aggressive drivers may use shorter time than average drivers use. Thus, a time-dependent uncertain edge weight that describes the distribution of the travel costs of traversing an edge better captures the real traffic conditions. We call this *time-dependent uncertain weight modeling*.

Vehicle tracking data, such as GPS data, is increasingly available, which makes it possible to instantiate uncertain weight models. Specifically, GPS records can be map-matched to edges in the underlying road network, upon which they reveal the real traffic conditions on the edges. By dividing the time of a day into different time periods of interest, we can assign uncertain edge weights to each edge in the road network for these time periods. This way, time-dependent uncertain edge weights are obtained using map-matched GPS records.

**Stochastic Path Planning:** Due to the widespread use of deterministic weight modeling, major online navigation services typically suggest a single path with the least travel cost (e.g., the fastest path) or \( k \) paths with the top-\( k \) least travel costs (e.g., the top-3 fastest paths). Although such services are useful, users may still benefit from richer information, such as the travel time distributions of the returned paths, which may help them make better travel plans.
1.3.2 Our Proposed Solution

As to further study the path planning problems under time-dependent uncertain setting, we formalize a problem called **time-dependent non-dominated stochastic path problem**.

First, based on findings from Section 1.2, we assign time-dependent uncertain travel costs to edges in the road network, and we use a discrete representation of random variables, namely histograms. We partition time into Periods of interest, and for each edge \( e \), we assign a histogram represents the uncertain travel cost in each time period of interest.

Second, we propose **stochastic dominance** to compare two random variables \( X \) and \( Y \). Here, \( X \) stochastically dominate over \( Y \) if

\[
\forall a \in \mathbb{R}^+ (CDF_X(a) \geq CDF_Y(a)) \land \\
\exists b \in \mathbb{R}^+ (CDF_X(b) > CDF_Y(b)),
\]

where \( CDF_X \) (\( CDF_Y \)) is the cumulative distribution function (CDF) of random variable \( X \) (\( Y \)). Given the histograms of paths \( P_1 \) and \( P_2 \) with their travel cost distribution CDFs represented as piece-wise linear functions \( F_1 \) and \( F_2 \), a **travel cost dominance** check is performed against \( F_1 \) and \( F_2 \). \( F_1 \) stochastically dominates \( F_2 \) if and only if \( \mathbb{R}^+ \) can be partitioned into \( N \) sub-domains \( D_1, D_2, \ldots, D_{N-1}, D_N \) such that

\[
\forall i \in [1,N] (x \in D_i \Rightarrow F_1(x) \geq F_2(x)) \land \\
\exists i \in [1,N] (x \in D_i \Rightarrow F_1(x) > F_2(x)).
\]

Thus, a Time-dependent Non-dominated Stochastic Path (TNSP) query takes as input source and destination vertices \( s \) and \( d \) and a starting time \( t \), and it returns a set of paths from \( s \) to \( d \) such that they are not stochastically dominated by any other paths starting at time \( t \).

We first propose an approach based on Dijkstra’s algorithm to solve the TNSP problem that enumerates every possible path between the source and destination and prunes paths that are stochastically dominated by another path. Our experiments show that the performance of this approach deteriorates significantly as the number of edges in a path grows, so it is not applicable to large road networks.

Contraction hierarchy CH [38] is a technique to speed up shortest-path routing where "contracted" versions of the road network are precomputed. We propose an advanced solution with time-dependent uncertain travel cost aware contraction hierarchies. For a road network \( G = (V, E) \), \( V \) and \( E \) denote the vertex and edge sets in \( G \). A linear combination of the different criteria is used to order the vertices by their importance in the road network, and a smaller order indicates that a vertex is less important in the road network. Next, vertices are contracted in the order of the vertex importance, and shortcut edges are added. Figure 1.4 shows an example of road network \( G \), where \( v_0 \) is being contracted. When contracting \( v_0, e_1 \) and \( e_2 \) are removed, and every path within a hop limit (a predefined parameter to limit the
search space) other than \( P = \langle v_2, v_0, v_3 \rangle \) between \( v_2 \) and \( v_3 \) is compared with \( P \). If no witness path (a path with a smaller travel cost than \( \langle v_2, v_0, v_3 \rangle \)) exists, a shortcut edge \( e_s = \langle v_2, v_3 \rangle \) is added if path \( \langle v_2, v_0, v_3 \rangle \) represents the path between \( v_2 \) and \( v_3 \) with minimum travel cost, and the travel cost of \( e_s \) is the sum of \( e_1 \)'s and \( e_2 \)'s travel costs. After all vertices are contracted, a shortcut edge set \( E^+ \) is obtained.

![Figure 1.4: Example of Vertex Contraction](image)

A modified bidirectional Dijkstra’s algorithm [45] is employed on \( G' = (V, E \cup E^+) \) to find the shortest path. The algorithm searches from the source vertex in one direction (upward) and from the destination vertex in the opposite direction (downward), and the searches only expand to the vertices with higher importance than the currently-visited vertex. If the shortest path exists, those two searches meet at a certain vertex \( v \).

Recalling the characteristics of our time-dependent uncertain edge weights, we build an uncertain contraction hierarchy for each time period for the road network, and the resulting contraction hierarchies together represent the **time-dependent uncertain contraction hierarchy** for the road network. Take the road network in Fig. 1.4 as an example again. When \( v_0 \) is being contracted, shortcut edge \( e_s \) dominates every path found between \( v_2 \) and \( v_3 \) within a hop limit.

We then propose our advanced approach based on time-dependent uncertain contraction hierarchies. Unlike CH, we perform single-directional search between given source-destination pairs at a starting time, and the search only explores vertices with higher vertex importance than the vertex is being visited. The search space is reduced substantially and the advanced approach can be used for road networks at national scale.

**Contributions:**

- We formalize the time-dependent non-dominated stochastic path planning problem.
- We propose the time-dependent uncertain weight modeling of a road network, which is able to better capture real traffic conditions.
- We develop a query processing method based on uncertain contraction hierarchies to efficiently solve the non-dominated stochastic path planning problem.
We report on comprehensive experiments that involve a substantial GPS data set. These offer insight into the efficiency, accuracy, and scalability of the proposed methods.

1.3.3 Key Findings from Empirical Study

To evaluate our methods that compute time-dependent non-stochastic path queries, we use the road networks of Aalborg (AA), North Jutland (NJ), and Jutland (JU) from OpenStreetMap[^1]. Our experimental results indicate that:

- Given a source and destination in a road network and a start time, our methods can find TNSP paths that are significantly different from each other.
- The query processing time of the baseline method increases rapidly when the distance between the source and destination grows, while the CH-based method can compute TNSP queries efficiently for road network at national scale.
- The number of TNSP paths does not grow quickly as the distance between the source and destination grows.

1.3.4 Future Work

It is of interest to take more travel cost types (e.g., GHG emissions, travel distance) into account to enable non-dominated multi-cost stochastic path queries. In addition, the current algorithms can also be extended to support personalized non-dominated stochastic shortest paths.

1.4 Context-aware, Personalized Routing

1.4.1 Background and Motivation

Travel in road networks is an important aspect of our lives, and a variety of navigation services exist that offer suggested routes when supplied with a source, a destination, and an optional departure time. Such services provide all users with the same routes, and they do not take into account a user’s context beyond possibly the user’s departure time. Specifically, navigation services often recommend shortest routes or fastest routes, where the travel times are derived from speed limits[^12] rather than from actual driving conditions, e.g., peak vs. off-peak traffic.

A personalized and context aware routing service has the potential to deliver routes that better match the preferences of drivers than do existing routing services. In particular, better routing may be achieved by the modeling of driver preferences and more thorough modeling of traffic characteristics, as summarized next.

[^1]: http://www.openstreetmap.org/
1.4. CONTEXT-AWARE, PERSONALIZED ROUTING

1.4.1 Multiple Criteria: When drivers choose routes, they may consider more than one criterion, e.g., travel distance, travel time, fuel consumption, toll cost, number of traffic lights. Routes that only consider one criterion, e.g., shortest routes or fastest routes, may not fully fulfill drivers’ needs.

1.4.2 Time-Dependent Uncertainty: Dynamic criteria such as travel time and fuel consumption are time dependent. For example, traversing a road during peak hours may take much longer than that during off-peak hours. Moreover, time dependent costs are generally uncertain. For instance, aggressive driving may consume more fuel than moderate driving, but may also reduce travel time. The uncertainty may also vary across time.

1.4.3 Context-Aware Driving Preferences: Preferences generally vary across drivers and a single driver’s preferences may also depend on the context.

1.4.2 Our Proposed Solution

We first categorize travel costs as static or dynamic, and we provide techniques to obtain accurate dynamic costs from trajectories. A static cost is represented by a deterministic value, while a dynamic cost is modeled as a dynamic cost function $F : T \rightarrow RV$, where $T$ is the time domain of interest, e.g., a day or a week, and $RV$ is the set of all possible random variables.

Given $N$ different travel cost types, for each type of cost, we maintain a function that assigns travel costs to all edges. To derive dynamic costs, we propose two approaches to instantiate dynamic cost functions of the edges in a road network based on trajectories from the road network, namely, a discrete approach (DA) and continuous approach (CA). Specifically, the discrete approach of instantiating a dynamic cost function partitions a period of interest (e.g., a week day) into fixed-length intervals. In contrast, the continuous approach is a time-decaying-based technique that instantiates cost functions while smoothing the distributions across time.

In different contexts, a driver has different driving preferences. We first introduce a concept called efficiency ratio and then propose three different strategies to model drivers’ driving behavior and to compute efficiency ratios. By clustering the efficiency ratios derived from a driver’s trajectories, we are able to identify a driver’s contexts.

Having found $k$ contexts for driver $d_m$, we then learn a driving preference vector $w$ for each context. We view the identification of a driving preference in a context as a classification problem, which can be solved by linear optimization. We prove that it is sufficient to consider only the personalized skyline routes when identifying driving preferences, and we propose an efficient algorithm to compute the personalized skyline routes. We utilize the personalized skyline routes to automatically generate positive and negative training examples for the classification problem; and we learn a driving preference vector by minimizing a judiciously designed objective function.

Finally, to provide personalized and context-aware routing services, we identify an appropriate context and apply the preference in the context to compute the personalized skyline routes for the users.
Contributions:

- We propose techniques that are able to derive time-dependent, uncertain travel costs from trajectories.
- We propose techniques that enable identification of a driver’s contexts.
- We present techniques for automatically generating training data using personalized skyline routes and learning context-aware driving preference based on the training data.
- We report on a comprehensive empirical evaluation in a realistic setting with a substantial GPS trajectory data set, thus providing insight into the design properties of the proposed techniques.

1.4.3 Key Findings from Empirical Study

We conduct experiments using a realistic setting with a substantial GPS trajectory data set to study the effectiveness and efficiency of the proposed techniques. Three important observations follow from the empirical studies.

- The continuous approach (CA) captures the dynamic travel costs (i.e., travel times and fuel consumption) more accurately than does the discrete approach (DA).
- The proposed context identification and driving preference learning methods are able to identify distinct contexts for each driver; and they make it possible to identify a driving preference for each context for each driver from the driver’s historical trajectories. As a result, they enable effective context-aware and personalized routing.
- The run-time of the context-aware and personalized routing is acceptable for on-line use.

1.4.4 Future Work

Two interesting research directions exist. First, in addition to the temporal aspect, it is of interest to consider other aspects (e.g., spatial and spatio-temporal aspects). Second, some drivers’ preferences are not fully captured, so it is of interest to consider other forms of driving preferences, e.g., non-linear preferences, instead of weighted-sum based linear preferences.

1.5 Dissertation Organization

The remaining chapters of the dissertation correspond to self-contained papers. The papers are unedited except for formatting changes.
1.5. DISSECTATION ORGANIZATION


- Chapter 3 corresponds to the paper, and it extends publication [57]: Yu Ma, Bin Yang, Christian S. Jensen: “Enabling Time-Dependent Uncertain Eco-Weights For Road Networks” (submitted for publication).

- Chapter 4 corresponds to the paper: Yu Ma, Bin Yang, Christian S. Jensen: “A Practical Approach to Routing With Time-Varying, Uncertain Edge Weights” (to be submitted).


This following papers were completed during my Ph.D. studies but are not included in the dissertation.


Part II

Publications
Chapter 2

Ecomark: Evaluating Models of Vehicular Environmental Impact

Abstract

The reduction of greenhouse gas (GHG) emissions from transportation is essential for achieving politically agreed upon emissions reduction targets that aim to combat global climate change. So-called eco-routing and eco-driving are able to substantially reduce GHG emissions caused by vehicular transportation. To enable these, it is necessary to be able to reliably quantify the emissions of vehicles as they travel in a spatial network. Thus, a number of models have been proposed that aim to quantify the emissions of a vehicle based on GPS data from the vehicle and a 3D model of the spatial network the vehicle travels in. We develop an evaluation framework, called EcoMark, for such environmental impact models. In addition, we survey all eleven state-of-the-art impact models known to us. To gain insight into the capabilities of the models and to understand the effectiveness of the EcoMark, we apply the framework to all models.

2.1 Introduction

Reduction in the greenhouse gas (GHG) emissions is crucial for combating global warming that has increasingly adverse effects on life on Earth. The European Union (EU) aims to reduce GHG emissions by 30% by 2020 [1], and the group of Eight (G8) plans a 50% reduction by 2050 [4]. Australia aims at a more ambitious target: an 80% reduction by 2050 [2]. China targets a 17% reduction over 2010 levels by 2015 [3].

The transportation sector is the second largest in terms of GHG emissions, trailing only the energy sector. In the EU, emissions from transportation account for nearly a quarter of the total GHG emissions [10]. Further, road transport generates more than two-thirds of transport-related GHG emissions and accounts for about one-fifth of the EU total emissions of carbon dioxide (CO₂), the major greenhouse
CHAPTER 2. ECOMARK: EVALUATING MODELS OF VEHICULAR ENVIRONMENTAL IMPACT

gas [9]. Therefore, reduction targets such as the above pose great challenges to the transportation sector in general and to vehicular transportation in particular.

In addition to improved design of vehicles and engines, eco-driving [19, 20] and eco-routing [49, 81] are simple yet effective approaches, which can achieve approximately 8–20% reduction in fuel consumption and GHG emissions from road transportation. Eco-driving targets eco-friendly driver behavior, e.g., accelerating moderately, maintaining an even driving speed, and avoiding frequent starts and stops, etc. Eco-routing recommends routes that aim to minimize fuel consumption and GHG emissions.

An important first step in reducing environmental impact is to be able to measure the impact. Thus, scientists in research areas such as energy engineering, civil engineering, and environmental science have proposed a range of models that aim to measure fuel consumption and GHG emissions [25, 35, 71]. These models consider a wide range of factors, e.g., vehicle speed and acceleration, different physical features of vehicles, and geometric information on the spatial network in which the driving occurs.

While nearly a dozen impact models exist, no comprehensive comparison of these models exists. Moreover, the utility of the models for eco-driving and eco-routing is also not well understood. This paper represents the first attempt at addressing these deficiencies by developing an evaluation framework, called EcoMark, for evaluating and comparing impact models using GPS trajectories and a 3D spatial network.

The use of GPS trajectories and a 3D spatial network bring the following benefits: (i) Many vehicles are equipped with GPS, and GPS data is plentiful and easy to collect. Thus, GPS data sets offer very good coverage of spatial networks, which enables Ecomark-based comparison of models to occur in a broader range of settings than in previous evaluations, where only limited selections of routes were considered [14, 70]. (ii) GPS trajectories are capable of providing the dynamic, travel-related information required by the models, e.g., velocities and accelerations of vehicles, and of reflecting the traffic conditions in a spatial network. (iii) A 3D spatial network captures both the lengths and the grades (degree of incline or decline) of road segments, which are factors that affect fuel usage and GHG emissions. Although several models take grades into account [23, 54, 81], only Tavares et al. [81] report on empirical studies with a 3D spatial network to measure the influence of grades on fuel consumption and GHG emissions.

To the best of our knowledge, this paper proposal of EcoMark and its subsequent application of EcoMark represents the first comprehensive study of the environmental impact of road transportation using GPS trajectories and a 3D spatial network.

The paper makes three contributions. First, a sophisticated evaluation framework is proposed that encompasses a 3D spatial network model. Second, a categorization and comparison of all eleven known models that can estimate fuel usage and GHG emissions is conducted. Third, comprehensive experimental studies are conducted on a half-year collection of GPS data from vehicles traveling in North Jutland, Denmark. Interesting findings obtained from the evaluation framework are discussed.
2.2. RELATED WORK

The remainder of this paper is organized as follows. Section 2.2 gives a comprehensive review of the state-of-the-art techniques for estimating fuel consumption and GHG emissions. Section 2.3 presents the EcoMark framework and also covers the modeling of a 3D spatial network. Section 2.4 categories the 11 models into instantaneous models and aggregated models and discusses them in detail. Section 2.5 reports on the application of EcoMark to the impact models. Conclusions are drawn in Section 2.6.

2.2 Related Work

The fuel consumed by a vehicle is influenced by multiple factors [15, 22, 24], such as vehicle technology (e.g., vehicle model and size, engine power, and type of fuel), vehicle status (e.g., mileage, age, and engine status), vehicle operating conditions (e.g., vehicle velocity and acceleration, power demands, and engine speed), driving behavior (e.g., aggressive driving), air conditions (e.g., atmospheric pressure, air humidity, and wind effects), road conditions (e.g., road grade and surface roughness), and traffic conditions (e.g., vehicle-to-vehicle and vehicle-to-control interactions). In general, different models consider different selections of these factors to compute fuel usage and GHG emissions.

Models for estimating fuel consumption or GHG emissions have been developed over the past thirty years and can be classified macroscopic and microscopic scale models [47]. Macroscopic models [17, 34–36] account for the total fuel consumed during an extended time period (e.g., a day, a week, or a year) when traveling in an extended region (e.g., a city or a state) [24].

Macroscopic models are suitable for applications where coarse estimation of environmental impact is desired. However, macroscopic models are unable to accurately estimate the environmental impact of a particular road segment traveled by an individual vehicle or of a particular driving operation (e.g., braking hard), which are of interest in eco-routing and eco-driving.

In contrast, microscopic models estimate the instantaneous fuel consumption or GHG emissions of individual vehicles at given time points (usually at seconds) using instantaneous velocities and accelerations. Some models utilize additional information, including vehicle status, vehicle operating conditions, and road conditions.

Microscopic models are further classified into three categories [24]. Emission map models [42] provide lookups in velocity-acceleration matrices and return corresponding emission values. Regression-based models [25, 54, 71] employ mathematical functions of second-by-second velocities and accelerations of a vehicle to predict instantaneous fuel consumption or GHG emissions. They do not consider physical features of vehicles. Load-based models [41, 60, 66, 72, 77] are the most comprehensive microscopic models, and they consider a wide range of parameters, such as second-by-second velocities and accelerations, grades of road segments, air conditions, engine maximum power, gear ratio, and engine power demands.
CHAPTER 2. ECOMARK: EVALUATING MODELS OF VEHICULAR ENVIRONMENTAL IMPACT

Microscopic models are often employed to evaluate the environmental impact of individual road segments on a spatial network and particular driving operations. The comprehensive load-based models generally offer the best estimates of fuel consumption. However, the required parameters are difficult to obtain for individual vehicles in a scalable manner. In contrast, regression-based models are fairly easy to apply because their input, e.g., instantaneous velocities and accelerations, can be obtained directly from GPS trajectories. EcoMark aims to evaluate the environmental impact of travel using models that do not require vehicle-specific factors. The details of the qualifying models are covered in Section 2.4.

Rakha et al. conduct a comparison of a regression-based model, VT-Micro, a load-based model, CMEM, and a macroscopic model, MOBILE6. The study finds that VT-Micro estimates fuel consumption more accurately than do the other two models. The study uses GPS data as input and considers the fuel consumption of a vehicle when it travels on highway routes and on arterial routes, respectively. The estimates obtained from VT-Micro and CMEM follow the same trend and indicate that choosing arterial routes generate less emissions than when choosing highway routes, whereas estimates obtained from MOBILE6 suggest the opposite.

To the best of our knowledge, EcoMark is the first work that performs a comprehensive comparison and analysis of fuel consumption models using GPS vehicle tracking data and a 3D spatial network, and with an emphasis on how the models considered can be utilized for eco-routing and eco-driving.

2.3 EcoMark Design

Following an overview of EcoMark, we cover the modeling and construction of a 3D model of a transportation network and describe the trajectories that are used in EcoMark.

2.3.1 EcoMark Overview

EcoMark is designed to evaluate state-of-the-art models of vehicular environmental impact in terms of fuel consumption and GHG emissions. It aims to provide an understanding of the utility of the impact models in relation to eco-driving and eco-routing and to offer insight into aspects such as which models can be used for identifying relationships between environmental impact and driver behavior, thus enabling eco-driving, and which models are suitable for assigning weights to road segments that capture environmental impact, thus enabling eco-routing.

Figure 2.1 depicts an overview of EcoMark. Three types of raw data are used in EcoMark: a set of GPS observations, a 2D spatial network, and a laser scan point cloud. A map matching module takes as input the set of GPS observations and the 2D spatial network. It outputs a set of map matched trajectories. A 3D spatial network generation module creates a 3D spatial network from the 2D spatial network...
2.3. **ECOMARK DESIGN**

and the laser scan point cloud. The trajectories and the 3D spatial network are fed into EcoMark as input data.

Unless stated otherwise, we use 2D to denote the latitude-longitude plane \((x, y)\) plane) and 3D to denote the latitude-longitude-altitude space \((x, y, z)\) space).

Road grades are generally difficult to obtain because the major maps available, e.g., OpenStreetMap [6], Google Maps [11], and Bing Maps [8], are 2D and lack grades of road segments. Thus, although some models [23, 54] take grades into account, this parameter has generally been set to zero in practice due to the unavailability of segment grade information.

In EcoMark, we use a 3D spatial network that provides grade information for all road segments. This network is constructed by using a laser scan point cloud for lifting a 2D spatial network. The use of laser point data yields a model with much higher accuracy than what can be obtained when using Shuttle Radar Topography Mission (SRTM) data [7], which has been used in the past. This is because the SRTM data contains only one altitude value for each \(30m \times 30m\) region, whereas the laser data we use contains one altitude value for each \(1m \times 1m\) region. The generation of the 3D spatial network is covered in Section 2.3.4.
In principle, the models supported in EcoMark take as input traffic and road information that can be obtained from GPS trajectories and a 3D spatial network, but do not require vehicle-specific factors. The supported models are categorized into instantaneous models and aggregated models. The instantaneous models take as input instantaneous (i.e., second-by-second) velocities and accelerations and output instantaneous fuel usage or GHG emissions. In contrast, the aggregated models take as input average velocities and output aggregated fuel usage or GHG emissions. The aggregated models can be applied at different aggregation levels, e.g., at the level of road segments or at the level of trajectories.

We use EcoMark to perform the following comparisons and analyses: (1) comparison and analysis of instantaneous models; (2) comparison and analysis of aggregated models; (3) aggregation of the instantaneous results, and comparison of them with the results obtained from the aggregated models; (4) comparison of the (instantaneous and aggregated) results with and without the use of road grades. Finally, meaningful conclusions, in terms of eco-driving and eco-routing, are drawn based on the sophisticated comparisons and analyses.

### 2.3.2 Trajectories

Since vehicle tracking using GPS is widespread and growing, GPS based vehicle tracking data is increasingly available and is thus an attractive source of vehicle movement data [69]. Thus, it is used in EcoMark.

A trajectory, denoted as \( T = (p_1, p_2, \ldots, p_X) \), is a sequence of GPS observations, where a GPS observation \( p_i \) specifies the location (typically 2D with latitude and longitude coordinates) and velocity of a vehicle at a particular time point. It is clear that instantaneous velocities are available from GPS trajectories, and instantaneous accelerations can be derived based on consecutive GPS observations.

Given a spatial network \( G \), a map matching algorithm [67] is able to associate each observation in a trajectory with a specific location on an edge in \( G.\mathcal{E} \). The map matching algorithm also enhances the accuracy of GPS trajectories by correcting inaccurate observations and filtering noisy observations.

An alternative to using GPS is to use roadside technologies, e.g., Bluetooth sensors and loop detectors, for the capture of vehicle velocities. However, only Bluetooth sensors may be able to link individual observations to specific vehicles, and the observations are much less frequent than what can be achieved with GPS. Thus, such velocities are less attractive for the estimation of the environmental impact of a vehicle.

### 2.3.3 Modeling a 3D Spatial Network

A spatial network captures both topological and geometric aspects of a transportation network in a certain region. In EcoMark, a 3D spatial network is defined as a directed, weighted graph \( G = (\mathcal{V}, \mathcal{E}, F, H) \), where \( \mathcal{V} \) and \( \mathcal{E} \) is the vertex set and edge set, respectively; and \( F \) and \( H \) are functions that record the geometric information of
vertices and edges, i.e., record the embedding of vertices and edges into geographical space.

A vertex $v_i \in V$ indicates a road intersection or an end of a road. $E$ is the edge set, and an edge $e_k \in E \subseteq V \times V$ is defined as a pair of vertices and represents a directed road segment connecting the two constituent vertices. For example, edge $e_k = (v_i, v_j)$ represents a road segment that enables travel from source vertex $v_i$ to target vertex $v_j$.

Function $F : V \rightarrow \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ takes as input a vertex and returns a 3D point. Function $H : E \times \mathbb{R} \rightarrow \mathbb{R}$ takes as input an edge and a value $x$, where $x$ indicates the distance along the edge from the source vertex to a point on the edge. The function outputs the grade of the point on the edge. In EcoMark, positive grades indicate uphill directions and negative grades indicate downhill directions.

Assume that $v_1$ and $v_2$ are the vertices of a road segment in a 3D spatial network, and $A$ is the grade inflection point on the road segment. Figure 2.2 shows the altitude-longitude projection of the road segment.

For edge $(v_1, v_2)$, the grade of the sub-edge from $v_1$ to $A$ is $\theta$, and the grade of the sub-edge from $A$ to $v_2$ is 0. Thus, $H((v_1, v_2), x) = \theta$ if $0 \leq x \leq |v_1, A|_{3D}$, where $| \cdot |_{3D}$ indicates the 3D Euclidean distance between two 3D points; $H((v_1, v_2), x) = 0$ if $|v_1, A|_{3D} \leq x \leq |v_1, A|_{3D} + |A, v_2|_{3D}$. Similarly, for edge $(v_2, v_1)$, $H((v_2, v_1), x) = 0$ if $0 \leq x \leq |v_2, A|_{3D}$; $H((v_2, v_1), x) = -\theta$ if $|v_2, A|_{3D} \leq x \leq |v_2, A|_{3D} + |A, v_1|_{3D}$.

### 2.3.4 Realizing a 3D Spatial Network

The 3D spatial network generation module augments a 2D spatial network with appropriate altitude information extracted from a laser scan point cloud, which contains a set of 3D points reflecting the surface of a certain region. An approximated surface of the region becomes available by transforming these 3D points into a TIN (Triangular Irregular Network) [68] surface. The altitude information of a 2D spatial network becomes available by projecting the 2D spatial network to the TIN surface, thus generating its corresponding 3D spatial network.

Specifically, we consider the region of North Jutland, Denmark in EcoMark. We obtain a 2D spatial network of North Jutland from OpenStreetMap, and we employ a
laser scan point cloud that covers North Jutland to generate a 3D spatial network of North Jutland. The detail of realizing the 3D spatial network is beyond the scope of EcoMark.

If GPS observations encompass altitude information, road grades can also be derived from GPS trajectories. However, GPS observations can be inaccurate, rendering derived grades less accurate than grades obtained using a laser scan point cloud.

2.4 Model Analysis

EcoMark covers eleven state-of-the-art models that estimate vehicular environmental impact. These models are categorized into instantaneous models and aggregated models as defined in Section 2.3.1 and are described in Sections 2.4.1 and 2.4.2, respectively.

All the models share the property that their input can be derived from GPS trajectories, e.g., velocities and accelerations, or can be obtained from a 3D spatial network, e.g., road grades. Values are not specified for some parameters below due to space limitations, but can be obtained from the original papers. To ease the following discussion, important notation is introduced in Table 2.1.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_t$</td>
<td>instantaneous velocity at time point $t$</td>
<td>$m/s$ or $km/h$</td>
</tr>
<tr>
<td>$a_t$</td>
<td>instantaneous acceleration at time point $t$</td>
<td>$m/s^2$ or $km/h/s$</td>
</tr>
<tr>
<td>$\theta_t$</td>
<td>road grade at time point $t$</td>
<td>degree or %</td>
</tr>
<tr>
<td>$f_t$</td>
<td>instantaneous environmental impact at time point $t$</td>
<td>$g/s$ or $mL/s$</td>
</tr>
<tr>
<td>$f$</td>
<td>environmental impact per unit distance</td>
<td>$g/km$ or $mL/km$</td>
</tr>
</tbody>
</table>

Table 2.1: Notation

2.4.1 Instantaneous Models

2.4.1.1 EMIT

The EMiissions from Traffic (EMIT) model [25] was proposed by Cappiello et al. and aimed to require simple input while still considering physical features of vehicles. EMIT assumes that (1) the road grade is 0; (2) the engine power required
2.4. MODEL ANALYSIS

for accessories, e.g., air conditioning, is 0; (3) no history effects, e.g., cold-start, are present; and (4) only hot-stabilized conditions are considered. It takes as input instantaneous velocity \( v_t (m/s) \) and acceleration \( a_t (m/s^2) \) and predicts the fuel consumption rate \( f_t (g/s) \) at time point \( t \).

\[
f_t = \begin{cases} 
\alpha + \beta \cdot v_t + \gamma \cdot v_t^2 + \delta \cdot a_t \cdot v_t & \text{if } P_{tract} > 0 \\
\alpha' & \text{if } P_{tract} \leq 0 
\end{cases}
\]

where \( \alpha, \beta, \gamma, \delta, \zeta \) and \( \alpha' \) are model-specific parameters \cite{25}, and \( P_{tract} \) is calculated according to the following equation.

\[
P_{tract} = A \cdot v_t + B \cdot v_t^2 + C \cdot a_t + M \cdot g \cdot \sin \theta_t \cdot v_t,
\]

where \( A, B, C \) and \( M \) are parameters related to physical vehicle features, \( g = 9.81 \) is the gravitational constant, and \( \theta_t \) is the road grade and is always set to 0 \cite{25}.

2.4.1.2 VT-Micro

VT-Micro, by Rakha et al. \cite{15, 71}, is a regression model that uses only instantaneous velocities and accelerations without considering the physical features of vehicles. VT-Micro makes the following assumptions: (1) the model is developed for light-duty vehicles and trucks; (2) the model only estimates fuel consumption and vehicle emissions for hot-stabilized conditions and does not consider the effect of vehicle start.

The model takes as input the instantaneous velocity \( v_t (km/h) \) and acceleration \( a_t (km/(h \cdot s)) \), and it estimates fuel consumption or emission rate \( f_t (mL/s \text{ or } g/s) \) at time point \( t \) as follows.

\[
f_t = \begin{cases} 
\exp(\sum_{i=0}^{3} \sum_{j=0}^{3} (K_{i,j} \cdot v_i \cdot a_j^i)) & \text{if } a_t \geq 0 \\
\exp(\sum_{i=0}^{3} \sum_{j=0}^{3} (L_{i,j} \cdot v_i \cdot a_j^i)) & \text{if } a_t < 0 
\end{cases}
\]

where \( K_{i,j} \) and \( L_{i,j} \) are model coefficients for accelerating \((a_t \geq 0)\) and decelerating \((a_t < 0)\) conditions, respectively.

2.4.1.3 MEF

MEF, by Lei et al. \cite{51}, is a regression-based model based on VT-Micro \cite{71} and POLY \cite{82}. It employs not only current velocity and acceleration, but also historical accelerations, to estimate emissions and fuel consumption.

At time point \( t \), MEF takes as input the current velocity \( v_t \) and the current acceleration \( a_t \), and also the historical (up to 9 seconds before \( t \)) accelerations, i.e., \( a_{t-i}, i = 1, ..., 9 \). The instantaneous fuel consumption or emissions rate \( f_t \) (g/s) at time point \( t \) is calculated as follows.

\[
f_t = \begin{cases} 
\exp(\sum_{i=0}^{3} \sum_{j=0}^{3} (\lambda_{i,j} \cdot v_i^4 \times \bar{a}_j^i)) & \text{if } \bar{a}_t \geq 0 \\
\exp(\sum_{i=0}^{3} \sum_{j=0}^{3} (\gamma_{i,j} \cdot v_i^4 \times \bar{a}_j^i)) & \text{if } \bar{a}_t < 0 
\end{cases}
\]
where \( \lambda_{i,j} \) and \( \gamma_{i,j} \) are model coefficients and \( \bar{a}_t \) is the composite acceleration given as 
\[
\bar{a}_t = \alpha \cdot a_t + (1 - \alpha) \cdot \sum_{i=1}^{9} \frac{a_t - i}{9}, \text{ where } \alpha \text{ is set to 0.5.}
\]

### 2.4.1.4 SP

Vehicle **Specific Power** (SP) is due to Jiménez-Palacios [54]. Instead of designing a model for estimating emission rates or fuel consumption directly, Jiménez-Palacios defines SP as the instantaneous power per unit mass of a vehicle. Jiménez-Palacios argues that SP is directly related to the vehicle engine load and thus to emission rates and fuel consumption. Thus, the model is claimed to be more reliable than the models solely based on velocities and accelerations.

At time point \( t \), SP takes as input vehicle velocity \( v_t \) (m/s) and acceleration \( a_t \) (m/s\(^2\)) as well as road grade \( \theta_t \) (%) and the headwind into a vehicle \( v_w \) (m/s). The SP value at time point \( t \), denoted as \( SP_t \), is defined as follows.

\[
SP_t = v_t \cdot (1.1 \cdot a_t + 9.81 \cdot \theta_t + 0.132) + 0.000302 \cdot (v_t + v_w)^2 \cdot v_t
\]

### 2.4.1.5 Joumard

Joumard et al. [46] carried out a study to identify the most important parameters that influence fuel consumption and emissions rates and that thus can be used to assess the design of traffic management systems on their impact on traffic pollution. The model, denoted as **Joumard**, assumes that: (1) travel occurs in urban conditions, and (2) the engine is hot. Joumard takes as input instantaneous velocity \( v_t \) (m/s) and acceleration \( a_t \) (m/s\(^2\)), and it computes the instantaneous fuel \( f_t \) as follows.

\[
f_t = v_t + v_t \cdot a_t,
\]

and Joumard et al. do not specify the unit of \( f_t \).

### 2.4.1.6 SIDRA-Inst

SIDRA, by Bowyer et al. [23], is a framework that includes four fuel consumption estimation models. SIDRA has been developed into commercial products\(^1\). The four models are designed in an increasing order of aggregation and integrate vehicle energy features (e.g., vehicle mass and drag force). The four interrelated models follow the same modeling framework, where a more aggregated model is derived from a more detailed model. Specifically, the SIDRA framework includes an instantaneous model, a four-mode element (i.e., acceleration, cruise, deceleration and idle) model, a running speed model, and an average travel speed model, denoted as SIDRA-Inst, SIDRA-4Mode, SIDRA-Running, and SIDRA-Avg, respectively.

**SIDRA-Inst**, the least aggregated model, takes as input instantaneous vehicle speed \( v_t \) (m/s) and acceleration \( a_t \) (m/s\(^2\)) and road grade \( \theta_t \) (%) at time point \( t \), and

\(^1\)http://www.sidrasolutions.com/
it computes the instantaneous fuel consumption \( f_t \) (mL/s) as follows.

\[
    f_t = \begin{cases} 
        0.444 + 0.09 \cdot R_t \cdot v_t + [0.05 \cdot 4a_t^2 \cdot v_t] \text{if } R_t > 0 \\
        0.444 \text{ if } R_t \leq 0 
    \end{cases}
\]

where \( R_t = 0.333 + 0.00108 \cdot v_t^2 + 1.2 \cdot a_t + 0.1177 \cdot \theta_t \) is the total tractive force required to drive the vehicle.

### 2.4.2 Aggregated Models

#### 2.4.2.1 Song

Song et al. [79] offer a model that aims to capture the effects of dynamic traffic conditions on fuel consumption and to evaluate the effects of different operational strategies in a transportation network. The model, denoted as \textit{Song}, is built on SP, but omits the road grade used in SP. Given a route \( e \), Song takes as input all instantaneous vehicle velocities \((m/s)\) and accelerations \((m/s^2)\) recorded on \( e \) and computes a relative fuel consumption indicator \( I_e \) for a light duty vehicle that traverses \( e \). This is done as follows.

\[
    I_e = \frac{\beta \cdot \sum_{t=1}^{T} \tau_t + T}{\gamma \cdot \sum_{t=1}^{T} v_t},
\]

where \( T \) is the total travel time (s) on route \( e \); \( \beta \gamma \) and \( \varepsilon \) are coefficients [79]; and \( \tau_t \) is a surrogate variable of vehicle specific power at time point \( t \), which equals \( SP_t' \) if \( SP_t' > 0 \), and equals 0 if \( SP_t' \leq 0 \), where \( SP_t' = v_t \cdot (1.1 \cdot a_t + 0.132) + 0.000302 \cdot v_t^3 \).

#### 2.4.2.2 Tavares

Tavares et al. [81] propose a route optimization method for minimizing the vehicle fuel consumption that is needed for the collection and transportation of municipal solid waste on Santiago Island. The cost function, denoted as \textit{Tavares}, is derived from COPERT [65] and is designed for estimating fuel consumption of heavy duty diesel vehicles. It takes into account the average velocity \( \bar{v} \) (km/h) on a road segment, the percentage of vehicle load \( LP \), and the road grade \( \theta \) (%). It computes the fuel consumption per unit distance \( \bar{f} \) (g/km) as follows.

\[
    \bar{f} = FCS \cdot LCF \cdot GrCF,
\]

where \( FCS=1068.4 \cdot \bar{v}^{-0.4905} \) indicates the basic fuel consumption; \( LCF=1 + 0.36 \cdot \frac{(LP-50)}{100} \) indicates the additional influence of vehicle weight; and \( GrCF=0.41 \cdot e^{0.18 \cdot \theta} \) indicates the additional influence of road grades.

#### 2.4.2.3 SIDRA-4Mode

\textit{SIDRA-4Mode} [23] is a four-mode elemental model for estimating fuel consumption when a vehicle drives on a road segment, making repeated stops and starts. The
driving can be modeled as a cruise-deceleration-idle-acceleration-cruise (CDIAC) cycle. In particular, the overall fuel consumption $F_s$ (mL) is calculated by $F_s = F_{c1} + F_d + F_i + F_a + F_{c2}$, where $F_{c1}$ (mL) and $F_{c2}$ (mL) are the fuel consumptions when a vehicle cruises at speed $v_1$ with distance $x_1$ and at speed $v_2$ with distance $x_2$, respectively; $F_d$ (mL), $F_i$ (mL), and $F_a$ (mL) are the fuel consumption when the vehicle is decelerating, idle, and accelerating, respectively.

Due to the space limitation, only the function for cruise, i.e., $F_c$, is covered here. Readers may refer to the original report [23] for the details of $F_d$, $F_i$, and $F_a$.

Cruising occurs when traveling between consecutive accelerations or decelerations. The function $F_c$ takes as input the average cruising speed $v_c$ (km/h) (allowing small fluctuations), the cruising distance $x_c$ (km), and the road grade $\theta$ (%), and it computes the cruising fuel consumption $F_c$ (mL) as $F_c = \bar{f}_c \cdot x_c$, where the cruising fuel consumption per unit distance $\bar{f}_c$ (mL/km) is defined as follows.

$$\bar{f}_c = \frac{1600}{v_c} + 30 + 0.0075 \cdot v_c^2 + 108 \cdot k_{E1} \cdot E_{k+} + 171.2 \cdot E_{k+}^2 + 10.6 \cdot k_{G} \cdot \theta,$$

where $E_{k+}$ is the marginal fuel consumption due to speed fluctuations and $k_{E1}$ and $k_{G}$ are calibration parameters [23].

### 2.4.2.4 SIDRA-Running

**SIDRA-Running** [23] is a more aggregated model derived from SIDRA-4Mode. The overall fuel consumption of a trip made by a vehicle includes the fuel consumed during the period of running and the fuel used when the vehicle is stopped briefly. The model requires as input $x_s$ (km), the total travel distance; $t_s$ (s), the total travel time; $\theta$ (%), the road grade; $v_i$ (km/h) and $v_f$ (km/h), the initial and final velocity during each positive acceleration, with the constraint that $v_i < v_f$; and $t_i$ (s), the total idle (stopped) time.

In cases where $v_i$, $v_f$, and $t_i$ are unknown, they can be estimated using given functions, or they can be replaced by other parameters [23].

The fuel consumption of the running mode $F_s$ (mL) is estimated as $F_s = F_i + \bar{f}_r \cdot x_s$, where $F_i = 0.444 \cdot t_i$ is the fuel consumption during idle periods and $\bar{f}_r$ (mL/km) indicates the average fuel consumption per unit distance excluding idle periods, which is computed as follows.

$$\bar{f}_r = \frac{1600}{v_r} + 30 + 0.0075 \cdot v_r^2 + 108 \cdot k_{E1} \cdot E_{k+} + 54 \cdot k_{E2} \cdot E_{k+}^2 + 10.6 \cdot k_{G} \cdot \theta,$$

where $v_r = \frac{3600 \cdot x_s}{(t_s-t_i)}$ is the average running speed (km/h) and $k_{E1}$, $k_{E2}$, and $k_{G}$ are calibration parameters [23].
### 2.4. MODEL ANALYSIS

#### 2.4.2.5 SIDRA-Avg

**SIDRA-Avg**, the average travel speed fuel consumption model, is the simplest and most aggregated model in the SIDRA framework, where only the average travel speed is required [23]. SIDRA-Avg takes as input the total travel distance $x_s$ (km) and the total travel time $t_s$ (s) including the time when stopped.

It is suggested that the model can only be used in urban road networks and that the average travel speed should be below 50 km/h. When the average speed exceeds 50 km/h, SIDRA-Running should be used instead [23]. The fuel consumption per unit distance $f_a$ (mL/km) is defined as follows.

$$ f_a = \frac{1600}{v_s} + 73.8, $$

where $v_s = \frac{3600 \cdot x_s}{t_s}$ is the average travel speed (km/h).

### 2.4.3 Summary

In the summary of models in Table 2.2 ✓ (×) indicates that a model considers (does not consider) a feature.

The inputs required by the models are obtainable from GPS trajectories (e.g., velocities and accelerations) or from a 3D spatial network model (e.g., grades), and therefore all models can be applied straightforwardly in most road transportation networks. Table 2.2 shows that instantaneous models require both velocities and accelerations as input, while most aggregated models take only velocities as input. The road grade is an optional input parameter in the instantaneous as well as the aggregated models. Studies exist that indicate the benefits of considering road grades in the models [81].

Some models, specifically EMIT, SP, Song, and the SIDRA family, explicitly combine physical vehicle features (indicated by the Physical Features column in the table) in the fuel consumption computation, whereas the remaining models, namely

<table>
<thead>
<tr>
<th>Models</th>
<th>Input</th>
<th>Physical Features</th>
<th>Absoluto Value</th>
<th>Vehicle Type</th>
<th>Model Year</th>
<th>Data Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>VT-Micro</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>Light</td>
<td>2004 U.S.</td>
</tr>
<tr>
<td>SP [24]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>Light</td>
<td>1999 U.S.</td>
</tr>
<tr>
<td>MEF [33]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>Light</td>
<td>2010 China</td>
</tr>
<tr>
<td>SIDRA-Inst</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Light, Heavy</td>
<td>1985 Australia</td>
</tr>
<tr>
<td>Song [79]</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>Light</td>
<td>2009 China</td>
</tr>
<tr>
<td>Tavares [81]</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>Heavy</td>
<td>2009 Cape Verde</td>
</tr>
<tr>
<td>SIDRA-4Mode</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>Light, Heavy</td>
<td>1985 Australia</td>
</tr>
<tr>
<td>SIDRA-Running</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>Light, Heavy</td>
<td>1985 Australia</td>
</tr>
<tr>
<td>SIDRA-Avg</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>Light, Heavy</td>
<td>1985 Australia</td>
</tr>
</tbody>
</table>

Table 2.2: A Summary of Models for Estimating Vehicular Environmental Impact
CHAPTER 2. ECOMARK: EVALUATING MODELS OF VEHICULAR ENVIRONMENTAL IMPACT

VT-Micro, MEF, Joumard, and Tavares, do not consider such features. No correlation exists between the type of model, i.e., instantaneous or aggregated model, and the use of physical features.

Most models predict an absolute impact value with units (indicated by the Absolute Value column). The units of the absolute impact values reported by instantaneous models are typically volume unit per time unit (e.g., mL/s) or mass unit per time unit (e.g., mg/s). For the aggregated models, volume (or mass) unit per distance unit, e.g., mL/km or g/km, are reported. In contrast to the majority of models, SP and Song provide merely a relative indicator of fuel consumption. Although Joumard is able to predict an “absolute” value, its unit is unclear in the description of the model [46]. Therefore, we regard the value as a relative indicator.

All the models have prerequisites on vehicle types, as listed in the Vehicle Type column. The models are also developed during different years and in different regions (as listed in the Model Year and Data Source columns). Therefore, the gathered traffic information and the measured fuel consumption that were used to calibrate the models may also be different. This may to some extent explain the use of different parameters in the models.

2.5 Empirical Studies

We conduct comprehensive empirical studies to compare and analyze the 11 models, in order to obtain insight into the properties of the different categories of models and how to apply the models to eco-driving and eco-routing.

2.5.1 Setup

The two main input data sources of EcoMark are GPS trajectories and a 3D spatial network, as described in Section 2.3.1. We use GPS data collected from 150 vehicles traveling in North Jutland, Denmark, during January to June 2007. The GPS data covers a variety of traffic conditions, e.g., peak and off-peak hour traffic, highway traffic and arterial road traffic. The sampling frequency is 1 Hz, which makes application to the instantaneous models easy.

We apply an existing map matching tool [67] along with a 2D spatial network of North Jutland, Denmark obtained from OpenStreetMap to the GPS data, from which we get a set of 52,084 trajectories, denoted as T.

The statistics of the trajectories are provided in Table 2.3 that lists the number of trajectories belonging to different categories. In the table, roadseg indicates the number of road segments traversed by a trajectory; avgvel indicates the average velocity of a trajectory; and duration indicates the total travel time of a trajectory. The units of the average velocity and the duration are km/h and minute(s), respectively. The corresponding 3D spatial network of North Jutland, Denmark is obtained as described in Section 2.3.4.
2.5. EMPIRICAL STUDIES

<table>
<thead>
<tr>
<th># total</th>
<th>52,084</th>
<th># roadseg ≤100</th>
<th>47,333</th>
</tr>
</thead>
<tbody>
<tr>
<td># avgvel ≤ 50</td>
<td>37,018</td>
<td># roadseg &gt;100</td>
<td>4,751</td>
</tr>
<tr>
<td># 50 &lt; avgvel ≤ 100</td>
<td>14,701</td>
<td># duration &lt; 30</td>
<td>44,544</td>
</tr>
<tr>
<td># avgvel &gt; 100</td>
<td>365</td>
<td># duration ≥ 30</td>
<td>7,461</td>
</tr>
</tbody>
</table>

Table 2.3: Statistics of Trajectories in $T$

### 2.5.2 Evaluating Instantaneous Models

Experiments are conducted to evaluate whether the instantaneous models show consistent estimation of environmental impact given the same trajectories. To quantify the consistency between two models $m$ and $n$, vector based cosine similarity is applied, as defined in Equation 2.1.

$$
	ext{consistency}(m, n) = \sum_{T \in T} \cos(L^{(m)}_T, L^{(n)}_T)
$$

where $L^{(m)}_T$ is a vector that contains the impact values of trajectory $T$ computed by model $m$ and function $\cos(\cdot, \cdot)$ computes the cosine similarity between two vectors.

The similarity between each two instantaneous models is shown in Table 2.4.

As an example, a trajectory $T_1$, with 2,375 GPS observations, is chosen to compare the behaviors of the instantaneous models. Figure 2.3(a) and (b) shows the instantaneous environmental impact from the 70th to the 100th seconds as computed by the models; Figure 2.3(c) and (d) shows the instantaneous velocities and accelerations during the same period. Figure 2.3 also indicates that the instantaneous environmental impact estimated by most models, except MEF, are highly correlated with accelerations.

Experiments are also conducted on the environmental impact caused by different driving behaviors, which in turn are indicated by different instantaneous velocities and accelerations. Four trajectories with distinct driving behaviors are chosen,
as shown in Table 2.5 whose velocities and accelerations are also plotted in Figure 2.4(c) and (d). The accelerations of $T_5$, whose values are around zero, are omitted to increase the readability of Figure 2.4(d).

EMIT shows high consistency with most instantaneous models (see Table 2.4) and thus is utilized to demonstrate the environmental impact of the four trajectories. As shown in Figure 2.4(a) and (b), the results indicate that: (1) environmental impact of moderate accelerations ($T_5$ and $T_6$) is much less than the impact of aggressive accelerations ($T_3$ and $T_4$); (2) driving with high velocities and aggressive accelerations yields higher environmental impact than when driving with low velocities and aggressive accelerations. For example, although the variations in the accelerations of
2.5. EMPIRICAL STUDIES

2.5.3 Evaluating Aggregated Models

Aggregated models are used to estimate environmental impact per unit length on road segments traversed by trajectories. A similar consistency comparison is conducted to gain insight into the similarity of each pair of aggregated models by applying Equation 2.1. As shown in Table 2.6, all models show similar estimation trends with similarities higher than 80%. Having similarities of at least 96.9%, Tavares, SIDRA-Running, and SIDRA-Avg are highly consistent with each other.

Trajectory $T_2$, which covers 114 road segments, is utilized as an example to show the behaviors of the aggregated models. The aggregated impact estimated by the aggregated models between the 50th to the 100th road segments is reported in Figure 2.5. In general, all the aggregated models behave similarly: the average velocity on a segment is inversely proportional to the environmental impact per unit length.

For example, the velocities on the 75th and 78th segments (see the two dips in $T_4$ is greater than those of $T_3$, $T_4$ generally has a lower environmental impact than does $T_3$.
CHAPTER 2. ECOMARK: EVALUATING MODELS OF VEHICULAR ENVIRONMENTAL IMPACT

Figure 2.5: Comparison of Aggregated Models on Trajectory $T_2$

<table>
<thead>
<tr>
<th>Model</th>
<th>Song (relative)</th>
<th>SIDRA-4Mode (mL/km)</th>
<th>SIDRA-Running (mL/km)</th>
<th>SIDRA-Avg (mL/km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tavares</td>
<td>86.4%</td>
<td>90.8%</td>
<td>96.9%</td>
<td>97.3%</td>
</tr>
<tr>
<td>SIDRA-4Mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIDRA-Running</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIDRA-Avg</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.6: Consistencies of Aggregated Models
2.5. **EMPIRICAL STUDIES**

![Graphs of EMIT and VT-Micro, SIDRA-Inst, Joumard, and SP and MEF aggregated impact](image)

**Figure 2.6: Aggregation of Instantaneous Models on Trajectory $T_2$**

Figure 2.5(d)) are lower than on other segments traversed by $T_2$, and thus the corresponding environmental impact per unit length on the two segments is higher. As it takes longer to traverse a unit length with a lower velocity, a higher environmental impact per unit length results.

Existing routing algorithms [56, 91] can be employed to enable eco-routing if the edge weights reflect the environmental impact of traversing the edges (i.e., road segments). Based on the findings reported in Figure 2.5 and Table 2.6, we argue that the aggregated models are appropriate for assigning eco-weights (i.e., environmental impact related weights) to road segments in order to enable eco-routing. We suggest the use of SIDRA-Running because the minimum input required, the total length and total travel time (or average velocity) of segments, are fairly easy to obtain.

Although the impact estimated by Song and SIDRA-4mode are more sensitive to changes in velocity than for the other aggregated models (see Figure 2.5), these two models require instantaneous travel information during computation. Therefore, they are not easy to use in scenarios where only average velocities are known. SIDRA-Avg and Tavares are not recommended because the former cannot be used when velocities are above 50km/h [23] and the latter is intended for heavy vehicles.
 Given a road segment, the impact estimated by SIDRA-Running is closely related to the average velocity. Thus, it is useful to determine whether different trajectories on the same road segment have similar average velocity. Thus, we study the 222 road segments that are frequently traversed by $T$. The absolute deviations of the average velocities on these road segments are reported in Figure 2.8(a). We find that on 86 segments, the absolute deviations of the average velocities are smaller than 2km/h, as seen in the first bar. Only on 3 segments are the absolute deviations of average velocities high, i.e., between 14km/h and 16km/h, as seen in the 8th bar. Figure 2.8(a) indicates that the average velocities on most road segments are similar, which renders it is possible to derive a single, meaningful eco-weight for each of such road segment.

The average velocities on some road segments vary significantly over time. In some cases, this is due to rush-hour congestion. We plot the absolute deviation during different periods of a particular road segment in Figure 2.8(b). Although the average velocities are quite different during the entire day (see the bar for All), the average velocities are similar during peak and off-peak periods (bars PEAK and OFFPEAK). Thus, time dependent eco-weights are suggested for such road segments, e.g., differ-

![Figure 2.7: Effect of Road Grades on Trajectory $T_7$](image-url)
ent eco-weights should be used for different periods.

For road segments where drivers behave significantly different, i.e., with large average velocity deviations and without clear peak and off-peak periods, ranges are suggested to be used as eco-weights. Alternatively, driver specific eco-weights should be considered.

2.5.4 Aggregation of Instantaneous Models Versus Aggregated Models

The aggregated environmental impact on a road segment can also be obtained by aggregating the instantaneous impact that is estimated by an instantaneous model on the road segment. In the following experiment, the aggregation is computed by summing the instantaneous impacts on the segment and dividing the sum by the length of the segment.

Trajectory $T_2$ is utilized to demonstrate the degree of consistency of the aggregated impact estimated by aggregating instantaneous impact and the aggregated impact estimated directly by the aggregated models.

As shown in Figure 2.6, the aggregations of VT-Micro, EMIT, and SIDRA-Inst behave similarly to the aggregated models. In particular, VT-Micro shows the most consistent results. In contrast, the aggregations of Joumard, SP, and MEF do not behave consistently with the aggregated models and do not show relationships with average velocities (see Figure 2.5(d)).

The experiment suggests that some instantaneous models are not suitable for aggregation in order to estimate the aggregated environmental impact and thus should not be used for assigning eco-weights to road segments. However, VT-Micro, EMIT, and SIDRA-Inst demonstrate the ability to provide reasonable aggregated environmental impact.
2.5.5 Effect of Road Grades

Experiments are carried out to evaluate the influence of varying road grades on environmental impact. A hilly route, traversed by Trajectory $T_7$, where the road segments have both positive and negative grades is, therefore, chosen—see Figure 2.9.

![Figure 2.9: Road Grades](image)

(a) For Instantaneous Models  (b) For Aggregated Models

Figure 2.9: Road Grades

Figure 2.7(a) and (b) plots the instantaneous impact estimated by SP and SIDRA-Inst with and without considering grades. The results demonstrate that positive (negative) road grades cause higher (lower) the environmental impact compared to the impact without considering grades.

Two aggregated models, SIDRA-4Mode and Tavares, are chosen to demonstrate the effect of road grades on the aggregated impact. As shown in Figure 2.7(c) and (d), the aggregated impact estimated by both models is higher for segments with positive grade than for segments with negative grade. Tavares is more sensitive to road grades. The difference between the impact with and without grades as estimated by Tavares is greater than for SIDRA-4Mode.

2.5.6 Empirical Findings

**Instantaneous Models and Eco-driving:** Instantaneous models estimate instantaneous environmental impact, e.g., second by second fuel usage, from instantaneous velocities and accelerations. Since instantaneous velocities and accelerations reflect driving behaviors, instantaneous models can be used to measure the environmental impact of different such behaviors. Instantaneous models, along with classical data mining methods, can be used to classify good and bad driving behaviors in terms of environmental impact, which in turn may suggest good driving behaviors to drivers. Thus, instantaneous models are appropriate for eco-driving applications.

**Aggregated Models and Eco-routing:** Aggregated models estimate environmental impact per unit length using average velocities or starting and ending velocities. Such impact can be utilized for assigning eco-weights to road segments, thus enabling eco-routing.
2.6. CONCLUSIONS

From this point of view, the aggregated models offer a foundation for eco-routing. In addition, instantaneous impact on a road segment predicted by an instantaneous model, e.g., VT-Micro, can be aggregated and then used as the eco-weight of the road segment.

We suggest employing SIDRA-Running to compute eco-weights for road segments because the least input required by this model can be obtained from GPS trajectories. We also suggest that a single eco-weight per road segment suffices in some cases, while time dependent weights or a range of weights should be considered in other cases, as presented in Section 2.5.3.

**Importance of 3D Spatial Networks:** Road grades substantially affect the environmental impact predicted by both instantaneous and aggregated models. Therefore, the use of 3D spatial networks benefits both eco-driving and eco-routing.

2.6 Conclusions

We develop EcoMark that evaluates models of vehicular environmental impact. Eleven state-of-the-art impact models are categorized into instantaneous models and aggregated models. The models are compared and analyzed based on a substantial collection of 1 Hz GPS trajectories and a 3D spatial network. The empirical study suggests that the instantaneous models are appropriate for eco-driving, while the aggregated models are helpful for eco-routing. The use of a 3D spatial network that records road grades benefits both eco-driving and eco-routing.

In future work, it is relevant to study how well the models predict actual environmental impact. Such a study is possible if vehicular CAN bus data that records the actual GHG emissions and fuel usage along with corresponding GPS observations are available.
Chapter 3

Enabling Time-Dependent Uncertain Eco-Weights For Road Networks

Abstract

Reduction of greenhouse gas (GHG) emissions from transportation is an essential part of the efforts to prevent global warming and climate change. Eco-routing, which enables drivers to use the most environmentally friendly routes, is able to substantially reduce GHG emissions from vehicular transportation. The foundation of eco-routing is a weighted-graph representation of a road network in which road segments, or edges, are associated with eco-weights that capture the GHG emissions caused by traversing the edges. Due to the dynamics of traffic, the eco-weights are best modeled as being time dependent and uncertain.

We formalize the problem of assigning a time-dependent, uncertain eco-weight to each edge in a road network based on historical GPS records. In particular, a sequence of histograms is employed to describe the uncertain eco-weight of an edge at different time intervals. Compression techniques, including histogram merging and buckets reduction, are proposed to maintain compact histograms while retaining their accuracy. In addition, to better model real traffic conditions, virtual edges and extended virtual edges are proposed in order to represent adjacent edges with highly dependent travel costs. Based on the techniques above, different histogram aggregation methods are proposed to accurately estimate time-dependent GHG emissions for routes. Based on a 200-million GPS record data set collected from 150 vehicles in Denmark over two years, a comprehensive empirical study is conducted in order to gain insight into the effectiveness and efficiency of the proposed approach.

3.1 Introduction

The greenhouse effect is due to the concentration of greenhouse gases (GHG) in the Earth’s atmosphere, which prevents heat from escaping into space. The combustion
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

of fossil fuel results in GHG emissions, and transportation is a prominent fossil fuels burning sector. Thus, reducing the GHG emissions from transportation is crucial in combating global warming.

Eco-routing is an easy-to-employ and effective approach to reducing GHG emissions from transportation. Given a source-destination pair, eco-routing returns the most environmentally friendly route, i.e., the route that produces the least GHG emissions [16, 40]. The literature reports that eco-routing can yield 8–20% reductions in GHG emissions from road transportation [39].

Neither the shortest nor the fastest routes generally have the least environmental impact [16]. Figure 3.1 shows an example of the shortest route, the fastest route, and the eco-route between source $A$ and destination $D$.

![Figure 3.1: Eco-Route, Fastest Route, and Shortest Route](image)

Vehicle routing generally relies on a weighted-graph representation of a road network, where the vertices and edges represent road intersections and road segments, respectively. The key to enabling effective eco-routing is to assign eco-weights to the edges that accurately capture the environmental costs (i.e., GHG emissions or fuel consumption) of traversing the edges. Based on the resulting weighted graph and the types of weights, e.g., single-value weights, time-dependent weights, or uncertain weights, existing routing algorithms [48, 59, 88, 90] can be applied to enable eco-routing.

A single-valued edge weight typically cannot fully capture the environmental cost of traversing an edge. For instance, while traversing an edge, aggressive drivers may generate more GHG emissions than average drivers. Thus, emissions vary across drivers, and an uncertain eco-weight that records the distribution of the cost of traversing an edge captures reality better. Further, eco-weights are generally time dependent, due to the temporal variation in traffic. For instance, during peak hours, traversing an edge normally produces more GHG emissions than during off-peak hours. As an example, Fig. 3.2(a) shows GHG emissions cost values observed on an edge in our road network during off-peak and peak hours, and Fig. 3.2(b) shows the
corresponding uncertain edge weights of the edge during off-peak and peak hours as histograms.

![Figure 3.2: Time-Dependent Uncertain Edge Weight of An Edge](image)

According to a recent benchmark on vehicular environmental impact models [39, 40], environmental costs of traversing edges can be derived from GPS data using vehicular environmental impact models. Based on such models, a previous study [57] offers a preliminary attempt at assigning time-dependent, uncertain eco-weights to edges. That study derives a time-dependent histogram for each edge to represent the eco-weight of each edge. That study makes the assumption that the histograms on different edges are independent and proposes a method to estimate the GHG emission distributions of a route at a given time using the eco-weights.

This paper makes three main contributions to extend and enhance the previous study [57]. First, by introducing virtual edges and extended virtual edges, we make it possible to capture the dependence among eco-weights of adjacent edges. Second, we propose several histogram aggregation methods that are able to estimate GHG emissions of routes based on the eco-weights of edges, virtual edges, and extended virtual edges. Third, experiments are conducted on a comprehensive GPS data set that provide insight into the efficiency and accuracy of the paper’s proposals.

The remainder of the paper is organized as follows. Section 3.2 reviews related work, and Section 3.3 covers preliminaries and formalizes the problem. In Section 3.4, the methods for building and using an Eco Road Network with histogram-based eco-weights is proposed, and Section 3.5 describes how to estimate GHG emissions using the Eco Road Network. Section 3.6 reports on the experimental results, and Section 3.7 concludes.

### 3.2 Related Work

Although much work has been conducted to enable time-dependent [31, 32] and stochastic [43] routing services in different application scenarios, existing propos-
T-drive [91] is the most relevant work to our study. T-drive aims at providing a fastest-route service based on travel-time weights learned from GPS records obtained from taxis. Our study differs from T-drive in several respects. First, due to its different focus, T-drive does not assign weights to road-network edges, but identifies so-called landmarks and assigns weights to landmark edges that connect pairs of landmarks. This setup makes the T-drive approach inappropriate for general-purpose routing because T-drive suggests sequences of landmark edges, not sequences of road-network edges because landmark edges typically correspond to multiple paths in the road network. In contrast, our study assigns weights to edges and thus provides a general-purpose foundation for stochastic routing. Second, T-drive uses a sequence of histograms as the weight of a landmark edge, where each histogram represents the distribution of travel times during an interval. When doing so, T-drive uses the same buckets for all histograms on a landmark edge. In contrast, our proposal is able to assign different buckets (based on distinct distributions) to the histograms on an edge during different intervals. For instance, more buckets are used for representing more complex distributions, e.g., the GHG emissions distributions during peak hours. Third, T-drive assumes that adjacent landmark edges are independent. In contrast, we consider the travel cost dependencies between adjacent edges, which better captures reality. The experimental studies confirm that our approach achieves better travel cost estimations for routes. Fourth, T-drive does not compute a travel cost histogram for a sequence of landmark edges, but instead computes a single value based on individual drivers’ optimism indices. Going beyond the T-drive study, we propose different histogram aggregation methods to aggregate the histograms of the edges in a route such that the travel cost histogram of a route can be computed. Thus, our approach provides a foundation for stochastic routing. Finally, we consider also eco-weights, not only time-weights.

A few recent studies also consider how to derive weights using GPS data. One study [89] covers the estimation of single-valued eco-weights for edges with infrequent or no GPS records. However, it is unable to capture time-dependence and uncertainty. Orthogonal to this study, we assume a setting where edges have considerable amounts of GPS records, and we focus on capturing detailed GHG emissions distributions during different intervals for the edges. Another recent study [87] concerns the update of near-future (e.g., the next 15-min or 30-min) eco-weights based on incoming real-time GPS data. In contrast, we capture time-dependent GHG emissions for longer periods, e.g., a day, a week, or a month, based on historical GPS data. As a result, our work is complementary to that study.

3.3 Problem Setting and Definition

We proceed to cover the problem setting and to formalize the problem.
3.3. PROBLEM SETTING AND DEFINITION

3.3.1 Time-Dependent Histograms

Given a multiset of cost values \( C \), the range of the cost values \( \text{Range}(C) \) is the set of non-duplicated values that occur in \( C \) \([44]\). The data distribution of the cost values in \( C \), denoted as \( \text{DD}(C) \), is a set of \( \langle \text{val}, \text{prob} \rangle \) pairs, where \( \text{val} \) indicates a value in \( \text{Range}(C) \), and \( \text{prob} \) is the number of occurrences of the value in \( C \) divided by the total number of values in \( C \). An example is shown as follows, where multiset \( C \) contains GHG emission values observed from an edge.

\[
C = \{\{5,8,10,20,15,10,20,20,34,28\}\};
\]

\[
\text{Range}(C) = \{5,8,10,15,20,28,34\};
\]

\[
\text{DD}(C) = \{\langle 5,0.1 \rangle, \langle 8,0.1 \rangle, \langle 10,0.2 \rangle, \langle 15,0.1 \rangle, \langle 20,0.3 \rangle, \langle 28,0.1 \rangle, \langle 34,0.1 \rangle \}.
\]

In particular, a histogram \( H = \langle (b_1,p_1), \ldots, (b_n,p_n) \rangle \) is a vector of \( \langle \text{bucket}, \text{probability} \rangle \) pairs, where a bucket \( b_i = [f_i,l_i] \) indicates a range of cost values, where \( f_i \) and \( l_i \) indicate the starting and ending values of the range. The buckets are disjoint, i.e., \( b_i \cap b_j = \emptyset \) if \( i \neq j \); and all elements in \( \text{Range}(C) \) belong to the union of the buckets, i.e., \( b_1 \cup \ldots \cup b_n \). The width of a bucket is defined as \( |b_i| = l_i - f_i \). If every bucket in a histogram has the same width, the histogram is an equi-width histogram. A probability \( p_i \) records the percentage of the cost values that are in the range indicated by \( b_i \). The sum of all probabilities is 1, i.e., \( \sum_{i=1}^{n} p_i = 1 \).

Next, two equi-width histograms \( H_1 \) and \( H_2 \) are isomorphic if they have the same number of buckets representing the same data range. However, the probabilities of corresponding pairs of buckets may still be different.

Given a time interval of interest \( TI \), a Time Dependent Histogram is a vector of \( \langle \text{period}, \text{histogram} \rangle \) pairs, where the time interval \( TI \) is partitioned into periods. Specifically, in a time dependent histogram \( \text{tdh} = \langle \langle TI_1, H_1 \rangle, \ldots, \langle TI_m, H_m \rangle \rangle \), period \( TI_i \) is a period in \( TI \), and histogram \( H_i \) is the histogram of the cost values observed in period \( TI_i \). The periods partition the time interval of interest, i.e., \( TI_1 \cup \ldots \cup TI_m = TI \).

3.3.2 Road Networks and Trajectories

An Eco Road Network (ERN) is a weighted, directed graph \( G = (V,E,F) \), where \( V \) and \( E \) are vertex and edge sets. A vertex \( v_i \in V \) models a road intersection or the end of a road, and an edge \( e_k = (v_i,v_j) \in E \) models a directed road segment that enables travel from vertex \( v_i \) to vertex \( v_j \). Function \( F: E \rightarrow \text{TDH} \) in \( G \) assigns time-dependent and uncertain eco-weights to edges in \( E \); and \( \text{TDH} \) is the set of all possible time dependent histograms.

A Trajectory \( \text{trj} = \langle r_1, r_2, \ldots, r_x \rangle \) is a sequence of GPS records. Each GPS record \( r_i \) specifies the location (typically with latitude and longitude coordinates) and velocity of a vehicle at a particular time \( r_i.t \). Furthermore, the GPS records in a trajectory are ordered based on their timestamps. Given the road network where the trajectories occurred, a GPS record in a trajectory can be mapped to a specific location on an edge in the road network using some map-matching algorithm \([67]\).
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

3.3.3 Problem Definition and Solution Framework

Given a set $TRJ$ of map matched trajectories in a road network $G' = (V, E, \text{null})$, the paper studies how to obtain the corresponding Eco-Road Network $G = (V, E, F)$. Specifically, the key task is to determine $G.F$, which assigns time dependent histograms to edges, based on trajectory set $TRJ$.

An overview of the framework that determines $G.F$ is shown in Fig. 3.3. The pre-processing module transforms the map matched trajectories into a set $TRR$ of traversal records of the form $\text{trr} = (e, t_s, tt, ge, \text{trj}_j)$. A traversal record $r$ indicates that edge $e$ is traversed by trajectory $\text{trj}_j$ starting at time $t_s$. The travel time and the GHG emissions of the traversal are $tt$ and $ge$, respectively. The travel times can be derived directly from the GPS records as the difference between the times of the first and last GPS records. Different vehicular environmental impact models [40] can be applied to compute the GHG emissions from the GPS records. After pre-processing, each edge $e_i$ is associated with a set of traversal records $\text{TRR}_i = \{\text{trr} \in \text{TRR}|r.e = e_i\}$.

![Figure 3.3: Framework Overview](image)

The ERN construction module builds initial time-dependent histograms for edges based on their traversal records. Maintaining the time-dependent histograms of all edges in a large road network may incur a large storage overhead. To reduce the overhead, approximation and compression techniques are employed to reduce both the number of $(\text{period}, \text{histogram})$ pairs in the time-dependent histograms and the
3.4. ERN Construction

We propose methods to generate an Eco-Road Network from traversal records.

3.4.1 Initial Time Dependent Histograms

An initial time dependent histogram is built for every edge $e_i \in E$ based on the traversal records associated with $e_i$, i.e., $R_i$. Given a time interval of interest $TI$, e.g., a day or a week, and the finest temporal granularity $\alpha$, e.g., 15 minutes or one hour, $TI$ is split into $\lceil \frac{TI}{\alpha} \rceil$ periods, where the $j$-th period $T_j$ is $[(j - 1) \cdot \alpha, j \cdot \alpha)$. For each $T_j$, an equi-width histogram $H_j$ is built based on the traversal records that occurred in the period, i.e., $R_i^{(j)} = \{r | r.e = e_i \land r.t_s \in T_j\}$.

To ease the following histogram compression operations, we make sure the initial histograms on edge $e_i$ are isomorphic. The initial histograms share the same range $[l, u]$ where $l$ and $u$ are the lowest and highest GHG emissions (or travel times) observed in $R_i$. Further, the same number of buckets $N_{bucket}$ is used for all histograms, where $N_{bucket}$ is a configurable parameter. Thus, $\lceil \frac{TI}{\alpha} \rceil$ isomorphic histograms are obtained for each edge, where each histogram has $N_{bucket}$ buckets.

Assuming $\alpha$ is set to 1 hour, Fig. 3.4(a) shows two isomorphic histograms during periods [8 a.m., 9 a.m.) and [9 a.m., 10 a.m.) for an edge in North Jutland, Denmark. The high similarity of the two histograms motivates us to compress them into one histogram with little loss of information. We proceed to show how to compress the histograms using histogram merging and bucket reduction. Our methods are configurable so that histogram accuracy can be controlled.

3.4.2 Histogram Merging

If two temporally adjacent histograms $H_i$ and $H_{i+1}$ represent similar data distributions, it is potentially attractive to merge the two histograms into one histogram $H$ that represents the data distribution for the longer period $T = T_i \cup T_{i+1}$.

Given two distributions, several techniques exist to measure their similarity, such as cosine similarity, the K-S test, and the $\chi$-square test. The simplicity and efficiency
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

of computing cosine similarity makes it appropriate for evaluating the similarity of two histograms.

To facilitate the use of cosine similarity, we treat a histogram as a vector of probabilities. A histogram \( H = \langle (b_1, p_1), \ldots, (b_n, p_n) \rangle \) has the vector \( V(H) = \langle p_1, \ldots, p_n \rangle \). Since the initial histograms are isomorphic and equi-width, they have the same number of buckets, and each bucket in a corresponding pair has the same range. Thus, all the vectors are isomorphic, meaning that they have the same number of dimensions, with each dimension representing the same entity, i.e., the probability in a particular sub-range. The similarity between two histograms is defined by Equation 3.1:

\[
sim(H_i, H_j) = \frac{V(H_i) \odot V(H_j)}{\|V(H_i)\| \cdot \|V(H_j)\|},
\]

where \( \odot \) indicates dot product between two vectors, \( \cdot \) indicates the product between two reals, and \( \|V\| \) indicates the magnitude of vector \( V \).

When the similarity of adjacent isomorphic histograms \( H_i \) and \( H_{i+1} \) exceeds a threshold \( T_{\text{merge}} \), they are merged into a new histogram \( \overline{H} \). The weight of \( H_i \) is \( W_i = \sum_{k} H_{i,c} \cdot p_k \), where \( H_{i,c} \) is the total number of cost values that are used to derive \( H_i \), which is equivalent to the number of traversal records in the \( i \)-th period. The probability value for the \( k \)-th bucket in \( \overline{H} \) is given by Equation 3.2:

\[
\overline{H}.p_k = H_i.p_k \cdot W_i + H_{i+1}.p_k \cdot W_{i+1}, \ \forall \ k \in [1, n]
\]

When merging two isomorphic histograms, the obtained histogram \( \overline{H} \) is isomorphic to \( H_i \) and \( H_{i+1} \). The probability given by the \( k \)-th bucket in \( \overline{H} \) is not just the average of \( H_i.p_k \) and \( H_{i+1}.p_k \). As the number of traversal records in the \( k \)-th buckets of histograms \( H_i \) and \( H_{i+1} \) may be different, we use the weighted average to construct the data probability of the \( k \)-th bucket in \( \overline{H} \), as shown in Equation 4.2. We also maintain \( \overline{H}.c = H_{i,c} + H_{i+1,c} \) so that the count of traversal records in \( \overline{H} \) is available for subsequent merging steps.
Given an initial time-dependent histogram \( tdh \) for an edge, a corresponding merged time-dependent histogram \( \overline{tdh} \) is computed iteratively. In each iteration, a pair of adjacent histograms with the highest histogram similarity is identified. If the similarity exceeds a user-defined threshold \( T_{merge} \), the two histograms are merged according to Equation 4.2, and the union of the two argument histograms’ periods becomes the period of the new histogram. The iteration terminates when \( T_{merge} \) exceeds the highest histogram similarity. For example, the two histograms for adjacent periods shown in Fig. 3.4(a) are merged into the single histogram shown in Fig. 3.4(b).

### 3.4.3 Bucket Reduction

Histogram merging reduces the numbers of histograms. Bucket reduction reduces the sizes of individual histograms, which is orthogonal to histogram merging.

Bucket reduction transforms a histogram \( H \) into a new histogram \( \hat{H} \) that approximates the data distribution represented by \( H \) using fewer buckets. \( \hat{H} \) is not necessarily equi-width, meaning that different buckets may have different widths. Histogram regression is conducted by merging two adjacent buckets. The range of the new bucket is the union of the range of two original buckets, and the probability of the buckets is the sum of the probabilities of the two original buckets. Thus, given a histogram \( H = ((b_1, p_1), \ldots, (b_i, p_i), (b_{i+1}, p_{i+1}), \ldots, (b_n, p_n)) \), after merging buckets \( b_i \) and \( b_{i+1} \), the new histogram is \( \hat{H} = ((b_1, p_1), \ldots, (b_{i-1}, p_{i-1}), (b_x, p_x), (b_{i+2}, p_{i+2}), \ldots, (b_n, p_n)) \), where \( b_x = b_i \cup b_{i+1} \) and \( p_x = p_i + p_{i+1} \).

The sum of squared error (\( SSE \)) is employed to measure the discrepancy between the original histogram \( H \) and the histogram after bucket reduction \( \hat{H} \). Since the error is only introduced by the buckets we merge and we merge only two adjacent buckets \( b_i \) and \( b_{i+1} \), the error introduced by this operation is given by Equation 3.3:

\[
SSE(H, \hat{H}) = \left( \frac{|H.b_i|}{|H.b_i| + |H.b_{i+1}|} \hat{H}.p_x - H.p_i \right)^2 + \left( \frac{|H.b_{i+1}|}{|H.b_i| + |H.b_{i+1}|} \hat{H}.p_x - H.p_{i+1} \right)^2, \tag{3.3}
\]

where \( |H.b_i| \) is the range of the \( i \)-th bucket in histogram \( H \), and \( H.p_i \) is the probability of the \( i \)-th bucket in histogram \( H \). The accuracy of the histograms is defined to be the deviation of the distribution described by the histograms from the distribution of the original data, and our goal is to achieve small deviations, which indicate small accuracy losses. Moreover, a smaller \( SSE \) indicates that \( \hat{H} \) achieves a smaller accuracy loss compared to the original histogram \( H \).

We consider a scenario where a storage budget (i.e., a number of buckets) for an edge is given, and where we need to decide how to merge the buckets in the histograms that represent the GHG emissions for different periods so that we maximize the overall accuracy. For example, rather than distributing the buckets uniformly, we may use higher (lower) bucket budgets for histograms representing peak hours (off-peak hours).
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

Given a merged time dependent histogram $\overline{tdh}$ of an edge and a reduction threshold $T_{red}$ indicating the total number of buckets available for the edge, Algorithm 1 describes how to obtain a time dependent histogram that meets the storage budget while achieving least accuracy loss. Note that for different edges, the reduction threshold $T_{red}$, i.e., the bucket budget, may be different. A simple heuristic is to assign higher bucket quotas to edges that have many merged histograms in their time dependent histograms after histogram merging. The number of buckets used for an edge $e$ is proportional to the number of histograms associated with $e$.

Algorithm 1: HistogramBucketReduction

Input:
Merged time-dependent histogram of edge $e$: $\overline{tdh} = \langle (T_1, H_1), \ldots, (T_m, H_m) \rangle$;
Bucket reduction threshold: $T_{red}$.

Output:
Final time-dependent histogram of edge $e$: $\overline{tdh}$.

1: while total buckets in $\overline{tdh}$ exceeds $T_{red}$ do
2: \hspace{1em} $\text{minSSE} \leftarrow \infty$;
3: \hspace{1em} for each histogram $\overline{H}_i$ do
4: \hspace{2em} for each adjacent buckets $\overline{H}_i.b_j$ and $\overline{H}_i.b_{j+1}$ in $\overline{H}_i$ do
5: \hspace{3em} Generate candidate histogram $\overline{H}_i'$ by merging $b_j$ and $b_{j+1}$ in $\overline{H}_i$;
6: \hspace{3em} if $SSE(\overline{H}_i, \overline{H}_i') < \text{minSSE}$ then
7: \hspace{4em} Record the pair buckets in $\text{MinPairBuckets}$;
8: \hspace{3em} $\text{minSSE} \leftarrow SSE(\overline{H}_i, \overline{H}_i')$;
9: \hspace{2em} Merge the buckets pair in $\text{MinPairBuckets}$;
10: return $\overline{tdh}$;

Algorithm 1 works iteratively. For each iteration, it linearly scans all adjacent buckets pairs and finds the pair that achieves the smallest $SSE$ to merge (lines 2–9). Note that to identify two buckets that need to be merged, every histogram in the given $\overline{tdh}$ has to be checked. This process terminates when the total number of buckets for the edge is below the reduction threshold $T_{red}$.

Alternatively, a priority queue $Q$ can be used, where an element is an adjacent bucket pair and the priority of the element is the $SSE$ value of merging the two adjacent buckets. We do not use such a priority queue because maintaining the priority queue is complex when a pair of adjacent buckets are merged into one bucket. For example, assuming that the highest-priority element is $p = \langle b_1, b_2 \rangle$, after merging $b_1$ and $b_2$ into a new bucket $b'$, if there exist elements containing $b_1$ or $b_2$ in $Q$, i.e., $e_1 = \langle b_0, b_1 \rangle$ and $e_1 = \langle b_2, b_3 \rangle$, these should be updated to $e'_1 = \langle b_0, b' \rangle$ and $e'_1 = \langle b', b_3 \rangle$. 
3.5 GHG Emissions Estimation

After histogram merging and bucket reduction, we obtain an ERN where each edge is associated with a compact time dependent histogram. Here, we study how to use the obtained ERN to estimate the GHG emissions of traversing a route.

Rather than estimating a single value for a traversal of a route, e.g., the expected GHG emissions, we estimate the GHG emissions distribution using histograms. This yields much more detailed information than a single value and is useful in many applications, e.g., stochastic route planning [53, 88] and probabilistic threshold-based routing [43].

The distribution of the GHG emissions of a traversal on a route is estimated based on the ERN. In particular, the distribution of the GHG emissions of the traversal, also represented by a histogram, is achieved by aggregating pertinent (w.r.t. the traversal time) histograms of sub-routes in the route, where a sub-route is an edge or a sequence of adjacent edges if the edges have highly dependent GHG emissions. In particular, if the starting traversal time of a sub-route with time-dependent histogram \( tdh = \langle (T_1, H_1), \ldots, (T_m, H_m) \rangle \) is \( t \), histogram \( H_t \) is selected for histogram aggregation if \( t \in T_i \).

3.5.1 Modeling Dependence Among Adjacent Edges

When aggregating the histograms of the edges in a route, we need to consider the dependencies of the GHG emissions distributions of adjacent edges.

3.5.1.1 Dependence Analysis

Most existing studies [27, 53, 64, 85] assume that the travel costs (e.g., travel times) of adjacent edges are independent. To evaluate this assumption, we conducted an empirical study on a collection of frequently traversed adjacent edge pairs. Specifically, we identified 82 edge pairs that each is traversed by at least 1,000 trajectories.

The GHG emissions distributions of two adjacent edges are modeled as two random variables \( X \) and \( Y \), and the normalized mutual information \( \text{NMI}(X, Y) \) is applied to quantify the dependency between \( X \) and \( Y \), as defined in Equation 3.4:

\[
\text{MI}(X, Y) = \sum_{y \in Y} \sum_{x \in X} p(x, y) \cdot \log \left( \frac{p(x, y)}{p(x)p(y)} \right)
\]

\[
\text{NMI}(X, Y) = 2 \cdot \frac{\text{MI}(X, Y)}{\text{ET}(X) + \text{ET}(Y)},
\]

where \( \text{MI}(X, Y) \) is the mutual information between \( X \) and \( Y \), which quantifies the mutual dependency between the two random variables, \( \text{ET}(X) \) denotes the entropy of random variable \( X \), and \( \text{NMI}(X, Y) \) is the normalized mutual information between \( X \) and \( Y \). The use of normalized values makes it easier to evaluate and visualize the degree of dependency between two random variables. We use \( \text{NMI}(X, Y) \) to represent the degree of GHG emissions dependency between two edges. The NMI
values are normalized to the range $[0, 1]$, and an NMI value of 0 means that the two edges are independent while 1 is the maximum NMI value and larger NMI values indicate higher dependency.

Figure 3.5 shows the percentage of edge pairs w.r.t. different ranges of NMI values, it shows the NMI values for both the scenarios where time dependence is considered (with time) and not considered (without time). It suggests that most adjacent edges tend to be independent, which complies with the independence assumption. However, some adjacent edges do have non-negligible dependencies, as indicated by the last two buckets in Fig. 3.5.

![Figure 3.5: NMI Study](image)

It is also of interest to further investigate the dependency between two edges when taking the times of the traversals into consideration. For each edge pair, we partition the time domain according to the intervals obtained from both edges’ corresponding time dependent histograms. For each partition, we compute NMI using only the trajectories that occurred in the partition. The results, also shown in Fig. 3.5, suggest that the GHG emissions dependencies between adjacent edges are reduced when we take into account the traversal times. However, some adjacent edges still have relatively high dependencies (shown by the last two buckets).

3.5.1.2 Virtual Edges

The above study shows that the independence assumption does not always hold. To better model dependencies and thus improve on the state-of-the-art, we introduce the notion of a virtual edge for each pair of dependent adjacent edges (i.e., edges whose NMI exceeds the dependency threshold $T_{dep}$). It is worth noting that using a smaller $T_{dep}$ results in more virtual edges being generated in the ERN and also results in more preprocessing. As for normal edges, time dependent histograms can be obtained that represent the distributions of GHG emissions and travel times for virtual edges.

Figure 3.6 shows an example with a simple road network, along with numbers of trajectories in the road network, shown in Table 3.1. For example, 200 trajectories traversed $e_4$ (the 5-th line in Table 3.1). After traversing $e_4$, 140 trajectories continued on $e_5$ (the 6-th line in Table 3.1).
Table 3.2 shows the GHG emissions dependencies between adjacent edges. For example, to check whether edges $e_4$ and $e_5$ are dependent, we use the traversal records from the 140 trajectories that traversed both $e_4$ and $e_5$ to compute the NMI. As shown in the last line of Table 3.2, $e_4$ and $e_5$ have a high NMI value (the default dependency threshold $T_{dep}$ is 0.2), and thus they are considered as dependent. A virtual edge $e_v$ representing $e_4$ and $e_5$ is created. Our empirical study in Section 3.6 investigates the effect of varying the $T_{dep}$ value. The time dependent histograms of the virtual edge $e_v$ is generated based on the 140 trajectories that traversed both $e_4$ and $e_5$. In contrast, we use all 200 trajectories that traversed $e_4$ to generate the time dependent histograms of edge $e_4$.

![Figure 3.6: Edges, Virtual Edges, and Extended Virtual Edges](image)

<table>
<thead>
<tr>
<th>Trajectory</th>
<th>Trajectory Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\langle e_1, e_2 \rangle$</td>
<td>40</td>
</tr>
<tr>
<td>$\langle e_1, e_3 \rangle$</td>
<td>60</td>
</tr>
<tr>
<td>$\langle e_1, e_2, e_3 \rangle$</td>
<td>5</td>
</tr>
<tr>
<td>$\langle e_1 \rangle$</td>
<td>200</td>
</tr>
<tr>
<td>$\langle e_4, e_5 \rangle$</td>
<td>140</td>
</tr>
<tr>
<td>$\langle e_4, e_6 \rangle$</td>
<td>50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Edge Pairs</th>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\langle e_1, e_2 \rangle$</td>
<td>0.2</td>
</tr>
<tr>
<td>$\langle e_2, e_3 \rangle$</td>
<td>0.25</td>
</tr>
<tr>
<td>$\langle e_4, e_6 \rangle$</td>
<td>0.05</td>
</tr>
<tr>
<td>$\langle e_1, e_4 \rangle$</td>
<td>0.03</td>
</tr>
<tr>
<td>$\langle e_4, e_5 \rangle$</td>
<td>0.23</td>
</tr>
</tbody>
</table>

### 3.5.1.3 Extended Virtual Edges

Next, it is possible that more than two adjacent edges exist such that each adjacent edge pair is dependent. We use an extended virtual edge to represent such edges. For instance, $e_1$ and $e_2$ are dependent, and so are $e_2$ and $e_3$. Thus, an extended virtual edge $e_{ext}$ is used to represent $e_1$, $e_2$, and $e_3$.

Intuitively, it is possible to compute a time dependent histogram for an extended virtual edges based on the trajectories that use the extended virtual edges. However, this approach becomes unattractive due to two reasons. First, as the number of edges in an extended virtual edge increases, the number of trajectories that use the extended virtual edge decreases quickly. Based on an analysis on more than 200 million GPS
records collected in Denmark over two years, we show that the average trajectory counts that cover \( n \) (where \( 1 \leq n \leq 5 \)) consecutive edges in Fig. [3.7].

\[
\begin{array}{c|c|c|c|c|c}
\text{Number of Consecutive Edges} & 1 & 2 & 3 & 4 & 5 \\
\text{Trajectory Counts} & 2500 & 2000 & 1500 & 1000 & 500 \\
\end{array}
\]

Figure 3.7: Trajectory Counts

If we compute time dependent histograms of extended virtual edges using the corresponding trajectories, we risk to obtain biased distributions and then perform over-fitting to small amounts of trajectories. For example, if we compute a time dependent histogram for \( e_{ext} \) in Fig. [3.6], we may be over-fitting to the 5 trajectories (the third line in Table [3.1]). Second, it is impractical to store the time dependent histograms for all extended virtual edges as this calls for excessive storage.

To contend with this problem, we propose a method that estimates GHG emissions for an extended virtual edge \( e_{ext} = (e_1, e_2, \ldots, e_N) \), based on all the virtual edges in \( e_{ext} \). Based on the trajectories that use a virtual edge \( e_v = (e_1, e_2) \) in the extended virtual edge, the joint probability of \( e_1 \) and \( e_2 \) is used to represent the correlation between the GHG emissions on \( e_1 \) and \( e_2 \), denoted as \( f_{e_1,e_2}(b^i_1, b^j_2) = Pr[G_{e_1} = b^i_1, G_{e_2} = b^j_2] = Pr[G_{e_2} = b^j_2, G_{e_1} = b^i_1] \), where \( b^i_1 \) is the \( i \)-th bucket in the histogram of the \( i \)-th edge \( e_i \), i.e., \( H_i \); thus, \( G_{e_i} = b^i_1 \) indicates that the GHG emissions on \( e_i \) are within the width of the \( i \)-th bucket in \( e_i \)'s histogram.

The conditional probability of GHG emissions on edge \( e_1 \) yields \( b^i_1 \) given the GHG emissions on edge \( e_2 \) is \( b^j_2 \) is given in Equation [3.5]

\[
f_{e_1|e_2}(b^i_1|b^j_2) = \frac{f_{e_1,e_2}(b^i_1, b^j_2)}{f_{e_2}(b^j_2)} \tag{3.5}
\]

The estimated GHG emissions of a route \( R = \langle e_1, \ldots, e_n \rangle \) is considered as a range and is computed using the estimation of GHG emissions ranges of all edges in \( R \), and \( R_m \) represents the first \( m \) edges in \( R \). By using one bucket from each of the \( n \) histograms, we get a bucket sequence \( BS = (b^1_1, b^2_2, \ldots, b^N_{n-1}, b^n_n) \). Specifically, operator + computes the union of two ranges, i.e., given two ranges \( b_1 = [l_1, r_1) \) and \( b_2 = [l_2, r_2) \), \( b_1 + b_2 = [l_1 + l_2, r_1 + r_2) \). Thus, the range of \( BS \) can be computed as \( |BS| = b^1_1 + b^2_2 + \ldots + b^N_{n-1} + b^n_n \), which is the sum of all the buckets in one bucket sequence. The probability of traversing \( R \) that yields \( |BS| \) is represented as
3.5. GHG EMISSIONS ESTIMATION

To get the GHG emissions histograms of $n$ adjacent edges with dependence, we have to enumerate every bucket sequence using one bucket from each edge’s histogram and compute the GHG emission of each sequence. If each of these $n$ histograms has $M$ buckets, the number of bucket combinations is $M^n$. Using the lower bound $L$ and the upper bound $U$ of the result histogram $H_R$, which are the sum of the lower bounds and the sum of the upper bounds of $n$ histograms, to construct the result histogram $H_R$ as an equi-width histogram of $M$ buckets, we distribute the probabilities obtained from all the $M^n$ bucket unions into the buckets in $H_R$. Equation (3.7) shows how the probabilities of bucket unions are distributed to a bucket $B$ in $H_R$. One way to construct the result histogram is to use the lower and upper bounds of the marginal histograms. These bounds can be used to create an equi-width histogram that consists of two buckets $B_1$ and $B_2$ whose ranges are $[55, 85)$ and $[85, 115)$. The marginal GHG emissions on all the edges are shown in Table 3.3. The joint GHG emissions probabilities of $(e_1, e_2)$ and $(e_2, e_3)$ are shown in Table 3.4. Table 3.3: Sample GHG Emissions

<table>
<thead>
<tr>
<th>Edge</th>
<th>GHG emissions: (bucket: probability)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bucket 1</td>
</tr>
<tr>
<td>$e_1$</td>
<td>(20,30):0.2</td>
</tr>
<tr>
<td>$e_2$</td>
<td>(20,30):0.5</td>
</tr>
<tr>
<td>$e_3$</td>
<td>(15,25):0.3</td>
</tr>
<tr>
<td>$e_4$</td>
<td>(10,20):0.4</td>
</tr>
</tbody>
</table>

Table 3.5 shows the bucket sequences of $R$ together with their GHG emission ranges and probabilities, where the resulting histogram $H_R$ is an equi-width histogram that consists of two buckets $B_1$ and $B_2$ whose ranges are $[55, 85)$ and $[85, 115)$. 

$$f_R(|BS|) = f_{R_{n-1}}(b_1^1, b_2^2, \ldots, b_{n-1}^p)$$

$$= f_{e_n|e_{n-1}}(b_n^p | b_1^1, b_2^2, \ldots, b_{n-1}^p) \cdot f_{R_{n-1}}(b_1^1, b_2^2, \ldots, b_{n-1}^p)$$

$$= f_{e_n|e_{n-1}}(b_n^p | b_{n-1}^p) \cdot f_{R_{n-1}}(b_1^1, b_2^2, \ldots, b_{n-1}^p)$$

$$= \ldots$$

$$\prod_{i=1}^{n-1} f_{e_i, e_{i+1}}(b_i^j, b_{i+1}^k)$$

$$\prod_{i=2}^{n-1} f_{e_i}(b_i^k)$$

Equation (3.6) shows how the GHG emissions histograms of $n$ adjacent edges with dependence, we have to enumerate every bucket sequence using one bucket from each edge’s histogram and compute the GHG emission of each sequence. If each of these $n$ histograms has $M$ buckets, the number of bucket combinations is $M^n$. Using the lower bound $L$ and the upper bound $U$ of the result histogram $H_R$, which are the sum of the lower bounds and the sum of the upper bounds of $n$ histograms, to construct the result histogram $H_R$ as an equi-width histogram of $M$ buckets, we distribute the probabilities obtained from all the $M^n$ bucket unions into the buckets in $H_R$. Equation (3.7) shows how the probabilities of bucket unions are distributed to a bucket $B$ in $H_R$. One way to construct the result histogram is to use the lower and upper bounds of the marginal histograms. These bounds can be used to create an equi-width histogram that consists of two buckets $B_1$ and $B_2$ whose ranges are $[55, 85)$ and $[85, 115)$. The marginal GHG emissions on all the edges are shown in Table 3.3. The joint GHG emissions probabilities of $(e_1, e_2)$ and $(e_2, e_3)$ are shown in Table 3.4. Table 3.3: Sample GHG Emissions

<table>
<thead>
<tr>
<th>Edge</th>
<th>GHG emissions: (bucket: probability)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bucket 1</td>
</tr>
<tr>
<td>$e_1$</td>
<td>(20,30):0.2</td>
</tr>
<tr>
<td>$e_2$</td>
<td>(20,30):0.5</td>
</tr>
<tr>
<td>$e_3$</td>
<td>(15,25):0.3</td>
</tr>
<tr>
<td>$e_4$</td>
<td>(10,20):0.4</td>
</tr>
</tbody>
</table>

Table 3.5 shows the bucket sequences of $R$ together with their GHG emission ranges and probabilities, where the resulting histogram $H_R$ is an equi-width histogram that consists of two buckets $B_1$ and $B_2$ whose ranges are $[55, 85)$ and $[85, 115)$.
Table 3.4: Joint GHG Emissions

<table>
<thead>
<tr>
<th>(20,30)</th>
<th>[20,30)</th>
<th>(30,40)</th>
<th>[30,40)</th>
<th>(15,25)</th>
<th>(25,35)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(20,30)</td>
<td>0.1</td>
<td>1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>(30,40)</td>
<td>0.4</td>
<td>0.4</td>
<td>0.1</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.5: GHG Emissions for Bucket Sequences

<table>
<thead>
<tr>
<th>Bucket Sequence</th>
<th>GHG Emissions Range</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>(20, 30), (20, 30), (15, 25)</td>
<td>[55, 85)</td>
<td>0.04</td>
</tr>
<tr>
<td>(20, 30), (20, 30), (25, 35)</td>
<td>[65, 95)</td>
<td>0.06</td>
</tr>
<tr>
<td>(20, 30), (30, 40), (15, 25)</td>
<td>[65, 95)</td>
<td>0.02</td>
</tr>
<tr>
<td>(20, 30), (30, 40), (25, 35)</td>
<td>[75, 105)</td>
<td>0.08</td>
</tr>
<tr>
<td>(30, 40), (20, 30), (15, 25)</td>
<td>[65, 95)</td>
<td>0.08</td>
</tr>
<tr>
<td>(30, 40), (20, 30), (25, 35)</td>
<td>[75, 105)</td>
<td>0.32</td>
</tr>
<tr>
<td>(30, 40), (30, 40), (15, 25)</td>
<td>[75, 105)</td>
<td>0.08</td>
</tr>
<tr>
<td>(30, 40), (30, 40), (25, 35)</td>
<td>[85, 115)</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Take $B_1$ as an example. Its probability to get its range can be computed as follows.

$$f_R(\{|B_1|\}) = \frac{[55, 85) \cap [55, 85)]}{[55, 85)} \cdot 0.04 + \frac{[55, 85) \cap [65, 95)]}{[55, 85)} \cdot 0.06$$

$$+ \frac{[55, 85) \cap [65, 95)}{[55, 85)} \cdot 0.02 + \frac{[55, 85) \cap [75, 105]}{[55, 85)} \cdot 0.08$$

$$+ \frac{[55, 85) \cap [75, 105]}{[55, 85)} \cdot 0.32$$

$$+ \frac{[55, 85) \cap [75, 105]}{[55, 85)} \cdot 0.08$$

$$= \frac{30}{30} \cdot 0.04 + \frac{20}{30} \cdot 0.06 + \frac{20}{30} \cdot 0.02 + \frac{10}{30} \cdot 0.08 + \frac{20}{30} \cdot 0.08$$

$$+ \frac{10}{30} \cdot 0.32 + \frac{10}{30} \cdot 0.08$$

$$= 0.31$$

3.5.1.4 Sub-Routes

A route $R$ can be split into a set of sub-routes, as described in Algorithm 2, where a sub-route is an edge, a virtual edge, or an extended virtual edge. Since we model adjacent edges with (high) dependencies as virtual edges and extended virtual edges, each adjacent sub-route pair is independent. The next step is to describe how to estimate the GHG emissions for a route given an ERN.
3.5. GHG EMISSIONS ESTIMATION

Algorithm 2: RouteToSubRoutes

Input:
A route $R = (e_1, e_2, \ldots, e_N)$;

Output:
A sequence of independent sub-routes $S$;

1: $i \leftarrow 1$
2: while $i \leq N$ do
3: if $i < N$ then
4: $j \leftarrow i + 1$
5: if $i = N$ or $e_i$ is not dependent on edge $e_j$ then
6: Insert $e_i$ into $S$
7: $i \leftarrow i + 1$
8: else
9: while $j \leq N$ and $e_{j-1}$ and $e_j$ are dependent do
10: $j \leftarrow j + 1$
11: if $|\langle e_i, \ldots, e_{j-1} \rangle| = 2$ then
12: Insert the virtual edge $e_v$ that represents edges $\langle e_i, \ldots, e_{j-1} \rangle$ into $S$
13: else
14: Insert the extended virtual edge $e_{ext}$ that represents edges $\langle e_i, \ldots, e_{j-1} \rangle$ into $S$
15: $i \leftarrow j$
16: Return $S$

3.5.2 Histogram Aggregation

Histogram Aggregation takes two histograms $H_1$ and $H_2$ and yields a histogram $H'$ that represents the aggregated GHG emissions for traversing both edges. The aggregation of $H_1$ and $H_2$ is also a histogram and is denoted as $H_{agg}$. Algorithm 3 shows how to perform a form of histogram aggregation that we call bucket-wise histogram aggregation because it operates on all combinations of two buckets from each input histogram iteratively. Recall operator $+$ denotes the union of two ranges, i.e., $[f_1, l_1] + [f_2, l_2] = [f_1 + f_2, l_1 + l_2]$. The buckets in $H_{agg}$ are constructed by combining all bucket pairs from $H_1$ and $H_2$, using one bucket from each histogram (Lines 2–4). As there might be overlaps between buckets in $H_{agg}$, we rearrange the buckets in $H_{agg}$ to combine two buckets with the same data range, and we split a data range if it contains the range of another bucket (line 5); thus, we obtain an equi-width histogram as the final result that contains buckets without overlap and duplicates.

To illustrate, consider two histograms $H_1 = \langle ([0, 2], 0.2), ([2, 4], 0.8) \rangle$ and $H_2 = \langle ([0, 2], 0.4), ([2, 4], 0.6) \rangle$. Before being rearranged, the aggregated result is $H_{agg} = \langle ([0, 4], 0.08), ([2, 6], 0.32), ([2, 6], 0.12), ([4, 8], 0.48) \rangle$; after being rearranged, the result is $H_{agg} = \langle ([0, 2], 0.04), ([2, 4], 0.26), ([4, 6], 0.46), ([6, 8], 0.24) \rangle$.

Alternatively, two other histogram aggregation methods, namely point-wise aggregation and median aggregation, can be used.
Algorithm 3: BucketWiseAggregate

Input:
Independent histograms $H_1$ and $H_2$;

Output:
Aggregated Histogram $H_{agg}$;

1: for all buckets $H_1[i]$ in $H_1$ do
2:   for all buckets $H_2[j]$ in $H_2$ do
3:      newBucket ← $(H_1[i].b, H_2[j].b, H_1[i].p, H_2[j].p)$
4:      add newBucket to $H_{agg}$
5:   Rearrange the generated buckets;
6:  Return $H_{agg}$;

Point-Wise Aggregation: A histogram $H$ is first transformed into a probability mass function, which is represented as a list $L$ of $(val, prob)$ pairs that reflect the data distribution (see Section 3.3.1) in histogram $H$. The values here are in the range between the minimum and maximum data values in $H$. The resolution of the resulting values is customized by changing the histogram aggregate parameter $T_{agg}$ that defines the finest granularity.

For example, consider $H_1$: if $T_{agg} = 1$, the transformed list is $(0.08)\langle 0.1, 1, 0.1 \rangle \langle 2, 0.4 \rangle \langle 3, 0.4 \rangle$; and if $T_{agg}$ is 2, the transformed list is $(1, 0.2) \langle 3, 0.8 \rangle$. After obtaining the lists $L_1$ and $L_2$ from histograms $H_1$ and $H_2$, the list $L'$ for the aggregation of $H_1$ and $H_2$ is given by Equation 3.8

$$L'[v] = \sum_{p_i} p_i, prob \cdot p_j, prob, (p_i \in L_1 \wedge p_j \in L_2 \wedge p_i, val + p_j, val = v) \quad (3.8)$$

Algorithm 4 shows how point-wise aggregation aggregates two histograms. To illustrate, consider lists $L_1 = (1, 0.2), (3, 0.8)$ and $L_2 = (2, 0.4), (4, 0.6)$ as obtained from $H_1$ and $H_2$ when $T_{agg} = 2$. Following all the iterations in Algorithm 4 (Lines 2–6), the result list $L'$ is $(0.08)\langle 3, 0.08 \rangle \langle 5, 0.44 \rangle \langle 7, 0.48 \rangle$. Initially, $(1, 0.2)$ in $L_1$ means that the probability of range $[1, 1 + T_{agg}] = [0, 2]$ is 0.2 and $(2, 0.4)$ in $L_2$ means that the probability of $[2, 4]$ is 0.4; so after aggregation, $(3, 0.08)$ in $L'$ means that the probability of $[3, 7]$ is 0.08. Based on the probability derived from $L'$, the result histogram $H'$ is $((3, 5), 0.04), ((5, 7), 0.26), ((7, 11), 0.7))$.

Median Aggregation: The performance of point-wise aggregation deteriorates quickly as the number of edges in a route grows, which makes it unattractive for long routes. By making changes to point-wise aggregation, we obtain an alternative method with better performance and reasonable accuracy. When we transform a histogram into a list of $(val, prob)$ pairs, median aggregation uses the median point of the bucket, instead of using all points in the bucket. For example, $H_1$ and $H_2$ are transformed into $L_1 = (2, 0.2), (4, 0.8)$ and $L_2 = (3, 0.4), (5, 0.6)$. The remaining steps of median aggregation are identical to those in point-wise aggregation. Thus, the result list $L'$ is $((5, 0.08), (7, 0.44), (9, 0.48))$. Similar to point-wise aggregation, $(2, 0.2)$ in $L_1$ means that the probability of range $[1, 3]$ is 0.2 and $(3, 0.4)$
3.5. GHG EMISSIONS ESTIMATION

Algorithm 4: PointWiseAggregate

**Input:**
- Independent histograms $H_1$ and $H_2$;
- Aggregate resolution: $T_{agg}$;

**Output:**
- Aggregated histogram $H'$;

1: $L_1 \leftarrow \text{Trans}(H_1, T_{agg})$, $L_2 \leftarrow \text{Trans}(H_2, T_{agg})$;
2: for all $p_i$ in $L_1$ do
3:     for all $p_j$ in $L_2$ do
4:         $L'[p_i.val + p_j.val] \leftarrow L'[p_i.val + p_j.val] + p_i.prob \cdot p_j.prob$;
5:     Generate the result histogram $H'$ that approximates the data distribution derived from $L'$;
6:     Return $H'$;

in $L_2$ means that the probability of $[2, 4)$ is 0.4; and after aggregation, $(5, 0.08)$ in $L'$ means that the probability of $[3, 7)$ is 0.08. Thus, the final histogram $H'$ derived from $L'$ is $\langle ([3, 5), 0.04), ([5, 9), 0.72), ([9, 11), 0.24) \rangle$.

### 3.5.3 GHG Emissions Estimation for Routes

Assume that a traversal uses a particular route at a particular time. The distribution of GHG emissions of this traversal is estimated by aggregating the pertinent histograms (w.r.t. the traversal time) of the independent sub-routes in the route. To choose the pertinent histogram for a sub-route $sr$, we need to know when the traversal of $sr$ starts. Since the travel times of the sub-routes before $sr$ are uncertain, the traversal starting time of $sr$ is also uncertain.

Take a traversal on route $R = \langle e_1, e_4 \rangle$ in Fig. 3.6 as a running example. Figures 3.8 and 3.9 show the distributions of the two edges’ GHG emissions and travel times, respectively. When starting at 8:58 a.m., the confidences that the traversal arrives at $e_4$ before and after 9 a.m. are both 0.5 according to Fig. 3.8(a). As the GHG emissions distributions on edge $e_4$ are different before and after 9 a.m., as shown in Fig. 3.9(b), histogram aggregation is done separately for the two periods. Thus, different arrival times at $e_4$ may result in the need to consider different GHG emissions histograms.

Based on the travel time on $e_1$, the aggregated GHG emissions histograms for traversing $R = \langle e_1, e_4 \rangle$ is represented as $H_1 = \langle ([10, 30), 0.1), ([30, 50), 0.35), ([50, 70), 0.4), ([70, 90), 0.15) \rangle$ with confidence 0.5 (entering $e_4$ before 9 a.m.), and $H_2 = \langle ([10, 30), 0.15), ([30, 50), 0.4), ([50, 70), 0.35), ([70, 90), 0.1) \rangle$ with confidence 0.5 (entering $e_4$ after 9 a.m.) using histogram aggregation. Finally, we merge $H_1$ and $H_2$ into the single histogram $\langle ([10, 30), 0.1125), ([30, 50), 0.3675), ([50, 70), 0.3875), ([70, 90), 0.1375) \rangle$. Algorithms 5 and 6 show in detail how we get the final result GHG emissions histogram.
To explain how time-dependent histogram aggregation is done, we introduce the notions of arrival time period and arrival confidence. Consider a route $R = \langle e_1, e_2 \rangle$. The travel time and GHG emissions costs of $e_1$ and $e_2$ are represented as time-dependent histograms. Next, if a trip from $e_1$ to $e_2$ starts at time $t$, an arrival time period $P$ indicates that entering $e_2$ at any time point $t_p \in P$ leads to a single travel time histogram associated with a single GHG emissions histogram for $e_2$. An arrival confidence associated with $P$ that indicates the confidence of entering $e_2$ during $P$. To illustrate, consider the running example in Fig. 3.6 and a traversal that starts from $e_1$ at 8:58 a.m. Table 3.6 illustrates the arrival time periods for entering $e_4$ and the corresponding arrival confidence.

Algorithm 5 computes the GHG emissions histogram(s) for a route $R$ at a given starting time $t_s$. The algorithm starts with the first sub-route in $R$ and calls Algorithm 6 iteratively to compute the GHG emissions histogram of $R$, from the first
3.5. GHG EMISSIONS ESTIMATION

Table 3.6: Time-Dependent Histogram Aggregation Example

<table>
<thead>
<tr>
<th>Arrival Time Period</th>
<th>Arrival Confidence</th>
<th>GHG emissions of ( e_4 )</th>
<th>Travel Time of ( e_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>[8:58 a.m., 9:00 a.m.)</td>
<td>0.5</td>
<td>( ([0, 20), 0.4), ([20, 40), 0.6) )</td>
<td>( ([1, 3), 0.4), ([3, 5), 0.6) )</td>
</tr>
<tr>
<td>[9:00 a.m., 9:02 a.m.)</td>
<td>0.5</td>
<td>( ([0, 20), 0.6), ([20, 40), 0.4) )</td>
<td>( ([1, 3), 0.4), ([3, 5), 0.6) )</td>
</tr>
</tbody>
</table>

sub-route to the last sub-route in \( R \). The intermediate result set \( G_R \) contains travel cost estimation tuples of the form \( (H_G, H_T, c) \). A tuple indicates that traversing the first \( i \) sub-routes \( \langle sr_1, \ldots, sr_i \rangle \) in \( R \) starting at \( ts \), the GHG emissions and travel time cost distributions are \( H_G \) and \( H_T \), with the confidence of getting \( H_G \) and \( H_T \) is \( c \). As shown in Figs. 3.8–3.9, when traversing a route at a given starting time, the potential travel time may overlap with multiple time intervals with different GHG emissions and travel costs histograms. Therefore, multiple histograms are needed to represent all the possible GHG emissions and travel cost distributions with corresponding confidences. Set \( G_R \) is initialized to contain a single travel cost estimation tuple of the GHG emissions and travel time cost histogram of the first sub-route \( sr_1 \) of \( R \) (Line 1). Taking the example in Figs. 3.8–3.9 if starting at 8:58 a.m., \( G_R \) is initialized to \( (\langle ([10, 30), 0.5), ([30, 50), 0.5) \rangle, \langle ([0, 2), 0.5), ([2, 4), 0.6) \rangle, 1) \). For each, Algorithm 5 (Lines 2–6) considers a sub-route \( sr_i \), aggregating its GHG emissions histograms and travel costs histograms with all the tuples in the current \( G_R \) (Lines 4–5). Using the confidence of the GHG emissions histograms in \( G_R \) as their weights and applying Equation 4.2 we merge them into a single histogram \( H_R \) as the final result (Line 7).

Algorithm 5: HistogramAggregateForRoute

Input:
- A route \( R \) of consecutive sub-routes \( \langle sr_1, \ldots, sr_n \rangle \);
- Starting time of the trajectory \( ts \);

Output:
- GHG emissions histograms for \( R \);

1: \( G_R \leftarrow \{ (H_{G,sr_1}, H_{T,sr_1}, 1) \} \);
2: for \( i \leftarrow 2 \) to \( n \) do
3: \( G_{tmp} \leftarrow \emptyset \);
4: for all \( g \in G_R \) do
5: \( G_{tmp} \leftarrow G_{tmp} \cup \) TimeDependentHistogramAggregate\( (sr_i, g.H_G, g.H_T, g.c) \);
6: \( G_R \leftarrow G_{tmp} \);
7: \( H_R \leftarrow Merge(G_R) \);
8: Return \( H_R \);

Algorithm 5 aggregates the GHG emissions and travel time cost histograms of
the \( n + 1 \)-st sub-route \( sr \) with those of the first \( n \) sub-routes in a route \( R \) assuming the traversal of \( R \) starts at time \( ts \). The algorithm first determines the time intervals in which the GHG emissions time-dependent histograms of \( sr \) that overlap with the range of \( H_T \) (line 1). For the \( i \)-th time interval with its corresponding histograms, we aggregate the GHG emissions histogram \( H_{G,i} \) with \( H_G \) and travel time histogram \( H_{T,i} \) with \( H_T \) using the aggregation method shown in Algorithm 5. The aggregated results represent the corresponding GHG emissions and travel time cost histograms if traversing \( sr \) in the \( i \)-th travel time interval, and the confidence of getting such histograms is the product of \( c_i \) and \( c \), where \( c_i \) is the confidence of traversing \( sr \) in the \( i \)-th time interval (Lines 3–7). Table 3.7 shows the intermediate results of the aggregation. In the example in Figs. 3.8–3.9, consider route \( R = \langle e_1, e_4 \rangle \). If the traversal starts from \( e_1 \) at 8:58 a.m., a candidate set \( G_{sr} \) of travel cost estimation tuples is returned. Thus, the final result is \( \langle ([10, 30], 0.1125), ([30, 50], 0.3675), ([50, 70], 0.3875), ([70, 90], 0.1375) \rangle \).

Algorithm 6: TimeDependentHistogramAggregate

**Input:**
- Sub-route \( sr \);
- GHG emissions histogram \( H_G \);
- Travel time histogram \( H_T \);
- Confidence \( c \);

**Output:**
- Candidate set \( G_{sr} \);

1: \( H_T \) overlaps with \( n \) intervals in the GHG emission time-dependent histograms of \( sr \);
2: \( G_{sr} \leftarrow \emptyset \);
3: for \( i \leftarrow 1 \) to \( n \) do
4: Compute the confidence \( c_i \) of starting in the \( i \)-th interval \( intvl_i \).
5: Aggregate \( H_G \) and the GHG emissions histogram of \( sr \) in \( intvl_i \), denoted as \( H_{G,i} \);
6: Aggregate \( H_T \) and the travel time histogram of \( sr \) in \( intvl_i \), denoted as \( H_{T,i} \);
7: \( G_{sr} \leftarrow G_{sr} \cup (H_{G,i}, H_{T,i}, c \cdot c_i) \);
8: Return \( G_{sr} \);

3.6 Empirical Study

We conduct a range of experiments to gain insight into the accuracy, efficiency, and storage properties of the proposed methods.
### Table 3.7: Time-Dependent Histogram Aggregation Result

<table>
<thead>
<tr>
<th>Arrival Time Period</th>
<th>Arrival Confidence</th>
<th>GHG emissions of $e_4$</th>
<th>Aggregated GHG emissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>[8:58 a.m., 9:00 a.m.)</td>
<td>0.5</td>
<td>(0.20, 0.4),</td>
<td>([10, 30), 0.1),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(20, 40), 0.6)</td>
<td>([30, 50), 0.35),</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>([50, 70), 0.4),</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>([70, 90), 0.15)</td>
</tr>
<tr>
<td>[9:00 a.m., 9:02 a.m.)</td>
<td>0.5</td>
<td>(0.20, 0.6),</td>
<td>([10, 30), 0.125),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(20, 40), 0.4)</td>
<td>([30, 50), 0.375),</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>([50, 70), 0.375),</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>([70, 90), 0.125)</td>
</tr>
</tbody>
</table>

### 3.6.1 Experimental Settings

We use a large GPS tracking data set containing more than 200 million GPS records collected at 1 Hz from 150 vehicles in Denmark from January 2007 to December 2008. A total of 802K traversal records are generated from the data set. We use the road network of Denmark from OpenStreetMap[1] To get the best map-matching, we extract edges from OpenStreetMap data with the finest granularity, with 414K vertices and 1,628K edges. We apply an existing map-matching tool [67] to match the GPS records to the road network, from which we get a trajectory set $TR$ of 62K trajectories. We merge adjacent edges into longer ones when they can be viewed as conceptually a single edge.

Figure 3.10 shows the distribution of the lengths of the edges in the road network and the routes derived from our GPS data.

![Figure 3.10: Distributions of Edge and Route Length](http://www.openstreetmap.org/)
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

Impact from GPS data [39], is applied to compute GHG emissions of of the road network edges from the trajectories. The time interval of interest $TI$ is set to a day: [0 a.m., 12 p.m.).

For each edge, we build time dependent eco-weight histograms. Our analysis indicates that the major roads in urban region, i.e., Aalborg in North Jutland, have good coverage of traversal records. When an edge is not covered by sufficient traversal records, a GHG emissions value $EH$ is derived based on the length and the speed limit of the edge, which can be obtained from OpenStreetMap. Thus, the edge is associated with a single histogram with only one bucket, indicating an $EH$ with probability $1$.

For each edge, we consider a baseline approach that uses isomorphic histograms with variable bucket width for different time intervals to represent the uncertain time-dependent weight of each edge. Thus, every edge is assigned an uncertain time-dependent weight in this baseline approach and no compression techniques are employed.

The values used for the finest temporal granularity $\alpha$, the merge threshold $T_{\text{merge}}$, the reduction threshold $T_{\text{red}}$, the dependency threshold $T_{\text{red}}$, and the aggregation threshold $T_{\text{agg}}$ for point-wise aggregation are shown in Table 3.8, where default values are shown in bold.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_{\text{bucket}}$</td>
<td>20, 40</td>
<td>bucket number of initial histograms</td>
</tr>
<tr>
<td>$T_{\text{merge}}$</td>
<td>0.9, 0.91, 0.92, 0.93, 0.94, 0.95, 0.96, 0.97, 0.98, 0.99</td>
<td>histogram merge threshold</td>
</tr>
<tr>
<td>$T_{\text{red}}$</td>
<td>35, 40, 45, 50, 55, 60</td>
<td>bucket reduction threshold</td>
</tr>
<tr>
<td>$T_{\text{dep}}$</td>
<td>0.15, 0.2, 0.25</td>
<td>edge dependence threshold</td>
</tr>
<tr>
<td>$T_{\text{agg}}$</td>
<td>1, 2, 3, 4, 5</td>
<td>point-wise aggregation threshold</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>15 mins, 30 mins, 1 hour</td>
<td>finest time granularity</td>
</tr>
</tbody>
</table>

The algorithms for histogram merging and bucket reduction are implemented in Python, which is well suited for scientific and numeric computations. A machine with a 16-core Intel Xeon 2GHZ CPU, 32GB main memory, and 4TB external memory is used.

3.6.2 Running Time Efficiency

Figure 3.11(a) shows the time needed to build equi-width histograms and to merge the histograms for an edge. The time increases as the number of GPS records associated with an edge increases. Thus, if an edge is covered by more GPS records, it takes longer to build the initial histograms. We show the running time when setting $T_{\text{merge}}$ to 0.95 and 0.98 in Fig. 3.11(a), and our experiments suggest that smaller
3.6. EMPIRICAL STUDY

$T_{\text{merge}}$ leads to less histogram merge time. Figure 3.12(a) depicts the effect of varying the number of buckets $N_{\text{bucket}}$ on the running time of histogram merging. The figure suggests that a smaller $N_{\text{bucket}}$ renders histogram merging more efficient. Figure 3.13(a) shows the running time of bucket reduction using 6,000 traversal records, which is the largest number of records that an edge has in our data set.

We further study the efficiency of aggregating histograms. We use a set of 1,012 routes that each is traversed by at least 1,000 trajectories. On average, a route covers 16 edges. If the NMI of two adjacent edges exceeds $T_{\text{dep}}$, we treat them as a virtual edge. Figure 3.14(a) shows the performance of our bucket-wise histogram aggregation method (BA) for routes in comparison to the baseline method, as well as results when virtual edge ($BA + VE$) and extended virtual edges ($BA + EVE$) are considered, where $T_{\text{dep}}$ value is set to 0.2. Similarly, Figs. 3.15(a) and 3.16(a) show the performance of the point-wise and median aggregation methods in comparison to the baseline. Recall that the dependence threshold $T_{\text{dep}}$ is configurable. To illustrate the impact of $T_{\text{dep}}$, Fig. 3.17(a) shows the performance of bucket-wise aggregation when varying dependence threshold $T_{\text{dep}}$ and also considering the result of not considering $T_{\text{dep}}$, where we assume the GHG emission distributions are completely independent between any adjacent edges. The results suggest that with a smaller $T_{\text{dep}}$, the histogram aggregation run-time is lower, as more virtual edges are generated when building the ERN. The average time to generate GHG emissions histograms for a single edge is 72 microseconds. It is acceptable as it is considered as one-off preprocessing. Moreover, the generation of a virtual edge, which corresponds to merging two histograms, is done in 9 microseconds.

![Figure 3.11: Histogram Merging Study](image)

3.6.3 Estimation Accuracy

To study the approximation accuracy of our histograms, we measure the distance between the original data distribution and the derived histogram representations, in-
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

![Graphs showing varying number of buckets of initial histograms](image)

(a) Run-Time
(b) Accuracy

Figure 3.12: Varying the Number Buckets of Initial Histograms

![Graphs showing bucket reduction study](image)

(a) Run-Time
(b) Accuracy

Figure 3.13: Bucket Reduction Study

![Graphs showing bucket-wise histogram aggregation study](image)

(a) Aggregation Run-Time
(b) Aggregation Accuracy

Figure 3.14: Bucket-Wise Histogram Aggregation Study
3.6. EMPIRICAL STUDY

(a) Aggregation Run-Time
(b) Aggregation Accuracy

Figure 3.15: Point-wise Histogram Aggregation Study

(a) Aggregation Run-Time
(b) Aggregation Accuracy

Figure 3.16: Median Histogram Aggregation Study

(a) Aggregation Run-Time
(b) Aggregation Accuracy

Figure 3.17: $T_{dep}$ Study (bucket-wise histogram aggregation method)
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS
FOR ROAD NETWORKS

Let $dd = \{(v_1, p_1), \ldots, (v_n, p_n)\}$ be the original data distribution in a period, where $v_i$ and $p_i$ indicate a value and its probability. The accuracy of a histogram in the period is defined by Equation 3.9.

$$Err(H, dd) = \frac{1}{n} \sum_{i=1}^{n} \frac{|p_i - \frac{H.p_k}{|H.b_k|}|}{max(p_i, \epsilon)},$$

(3.9)

where the $k$-th bucket in $H$ contains $v_i$, i.e., $v_i \in H.b_k$, and $H.p_k$ is the total probability of the $k$-th bucket, and $|H.b_k|$ is the width of the $k$-th bucket in $H$. We use a constant $\epsilon = 0.001$ to avoid fluctuations caused by small probabilities, where $\epsilon$ is the smallest probability we consider in the our ERN. This accuracy metric captures the relative accumulative deviation from the original distribution.

Figure 3.11(b) shows average $Err$ values of the initial equi-width histograms and the histograms after merging for varying merge thresholds. The initial equi-width histograms achieve the least accuracy loss, and the merged histograms achieve improved accuracy as the merge threshold increases. We also evaluate the accuracy of bucket reduction. We choose two sets of merged histograms with merge thresholds 0.95 and 0.98, respectively. Moreover, Fig. 3.12(b) shows the impact of varying the number of buckets $N_{bucket}$ on the accuracy of histogram merging. The figure suggests that we can achieve a lower histogram merge loss by assigning more buckets to the initial histograms at the cost of run time. Figure 3.13(b) shows that the accuracy increases as the reduction threshold increases.

Next, we consider the 1,012 trajectories from Section 3.6.3 to evaluate the accuracy of histogram aggregation. We split the trajectories into two sets: training trajectories (from the first 18 months, 43K trajectories) and testing trajectories (from the last 6 months, 19K trajectories). For each route, we aggregate the histograms
from the ERN to estimate its GHG emissions histogram. We then generate ground-truth GHG emissions histograms for the route in each time interval without any data compression using the testing trajectories. The accuracy of histogram aggregation is defined as the histogram similarity \((HSimilarity)\) between the estimated histogram and the ground-truth histogram, using Equation 3.1. Figure 3.14(b) shows the accuracy of bucket-wise histogram aggregation \((BA)\) with varying the number of edges in a route, as well as the accuracy when virtual edges \((BA + VE)\) and extended virtual edges \((BA + EVE)\) are taken into account, and it also includes the baseline method. Similarly, Figs. 3.15(b) and 3.16(b) show the accuracy of point-wise and median aggregation and the baseline method.

Taking bucket-wise histogram aggregation as an example when virtual edges and extended virtual edges are considered, Fig. 3.17(b) shows the impact of the dependence threshold \(T_{dep}\) on aggregation accuracy and it also covers the case where \(T_{dep}\) was not considered. The figure indicates that a smaller \(T_{dep}\) yields more virtual edges and extended virtual edges, which better capture the traffic patterns, and thus better GHG emissions estimation accuracy is achieved. Figures 3.17(a) and (b) show that we can get estimation accuracy improvements by modeling adjacent edges with high dependency as virtual edges and extended virtual edges.

### 3.6.4 Storage Consumption

We next evaluate the storage savings that can be obtained by using histogram merging and bucket reduction. A bucket in a histogram requires two integer values (i.e., 8 bytes) to indicate the lower and upper bound of the bucket and a double (i.e., 8 bytes) for the bucket probability.

We introduce the memory compression ratio \(MCR\) to measure the storage reduction. In particular, the \(MCR\) for histogram merging is computed as \(MCR_m = \frac{M_{init} - M_{merge}}{M_{init}}\), where \(M_{init}\) and \(M_{merge}\) represent the storage required to represent the initial time dependent histograms and the merged histograms. The \(MCR\) for bucket reduction is computed as \(MCR_r = \frac{M_{merge} - M_{redu}}{M_{merge}}\), where \(M_{redu}\) represents the storage required to represent the histograms after bucket reduction based on merged histograms.

Figure 3.19(a) shows that when the merge threshold is set to 0.9, the storage required by the initial histograms can be reduced by 85%. When the merge threshold is 0.98, the reduction is 75%. Recall that the accuracy when using this merge threshold is quite close to that of the initial histograms (see Fig. 3.11(b)). We fix two sets of merged histograms (with merge thresholds 0.95 and 0.98) and observe the \(MCR\) with varying reduction thresholds. Figure 3.19(b) shows that bucket reduction can further reduce the required storage: smaller thresholds achieve better \(MCR\).

Figure 3.20 reports the average storage required for a single edge in order to achieve different accuracies for our different methods (\(T_{dep}\) not considered, \(VE\) used, \(EVE\) used) and the baseline method. The figure shows that to achieve a higher accuracy (i.e., a smaller \(Err\) value), more storage is required.
CHAPTER 3. ENABLING TIME-DEPENDENT UNCERTAIN ECO-WEIGHTS FOR ROAD NETWORKS

3.6.5 Summary

Based on our experimental results, the recommended parameter settings for $T_{\text{merge}}$, $T_{\text{red}}$, and $T_{\text{dep}}$ are 0.95, 50, and 0.2, respectively. With these settings, each edge requires on average 3.98 histograms and about 0.61 KB storage space. Assuming there is sufficient GPS data for all edges in the Denmark road network, to achieve the accuracy of the initial histograms without any compression, the storage usage of the
compact histograms is 2.82\textit{GB}. By using the default settings, the storage usage is 0.50\textit{GB} for the ERN for the Denmark road network. This suggests that our compact representation of time-dependent histograms reduces the storage substantially while maintaining good accuracy.

To sum up, better estimation accuracy can be achieved by using higher settings for $T_{\text{merge}}$ and $T_{\text{red}}$ and a lower setting for $T_{\text{dep}}$. This comes at the cost of more pre-processing time and higher storage consumption, and our recommended settings are chosen by taking the tradeoffs among all the different thresholds into consideration.

\section{Conclusions}

We present techniques capable of using a large trajectory data set for assigning eco-weights to the edges in a road network. The resulting Eco-Road Network, which assigns compact and time-dependent, uncertain eco-weights to edges, provides the key foundation for enabling eco-routing. We also study the correlations between the GHG emissions from adjacent edges, and we propose time-dependent GHG emissions distributions estimation techniques for routes. The accuracy and compactness of the proposed techniques are evaluated based on a 2-year GPS vehicle tracking data set. The experimental study shows that our method is able to save up to 20\% storage space in comparison with the baseline method while providing the same accuracy.

In future work, it is of interest to explore advanced routing algorithms that can fully utilize the time-dependent, uncertain eco-weights, e.g., to compute probabilistic eco-routes. Additionally, using an inverted approach that assigns a time dependent histogram based eco-weight to a group of edges that have similar travel cost distributions may result in further storage space reductions.
Chapter 4

A Practical Approach to Routing With Time-Varying, Uncertain Edge Weights

Abstract

It is becoming increasingly possible to capture travel costs such as travel time and greenhouse gas emissions associated with the movement of vehicles in road networks. Such costs vary over time and are inherently uncertain, due to factors such as varying traffic volumes, the weather, and different driving styles among drivers. This paper presents a practical approach to routing in a road network with time-varying, uncertain edge weights derived from GPS data from vehicles traveling in the road network. Specifically, we define and solve the time-dependent, non-dominated stochastic routing problem that finds all paths between a source-destination pair at a given start time whose travel costs are not stochastically dominated by any other path. This stochastic routing problem is a natural generalization of standard routing to the paper’s setting. We extend contraction hierarchies by exploiting properties of the problem setting, thus enabling efficient stochastic routing. Experimental studies with a substantial GPS data set suggest that the approach fits the setting and is capable of enabling efficient stochastic routing.

4.1 Introduction

Path planning is an indispensable part of many location-based services and is being used increasingly as Internet-enabled mobile devices continue to proliferate. Although useful route planning methods do exist, to fulfill user needs, we still need to address the following two challenges.

**Time-Dependent Uncertain Travel Costs:** Path planning relies on a weighted graph representation of a road network, where the weight of an edge (i.e., a road segment) refers to the travel cost of traversing the edge. Most existing work uses deterministic weight modeling, where the travel costs of traversing edges (i.e., edge
weights) are modeled as deterministic values. For instance, several major on-line service providers (e.g., Google Maps, Bing Maps) employ the lengths of road segments divided by the speed limits of the road segments as the travel time based edge weights [92]. However, deterministic weight modeling does not accurately capture the traffic conditions in road networks. On the one hand, the cost to traverse an edge may vary during a day, i.e., the average travel time during peak hours is higher than that of off-peak hours. On the other hand, while traversing the same edge, aggressive drivers may use shorter time than average drivers use. Thus, a time-dependent uncertain edge weight that describes the distribution of the travel costs of traversing an edge better captures the real traffic conditions. We call this time-dependent uncertain weight modeling.

Vehicle tracking data, such as global positioning system (GPS) data, is increasingly available, which makes it possible to instantiate uncertain weight models. Specifically, GPS records can be map-matched to edges in the underlying road network, upon which they reveal the real traffic conditions on the edges. By dividing the time of a day into different time periods of interest, we can assign uncertain edge weights to each edge in the road network for these time periods. This way, time-dependent uncertain edge weights are obtained using map-matched GPS records.

**Stochastic Path Planning:** Due to the widespread use of deterministic weight modeling, major online navigation services typically suggest a single path with the least travel cost (e.g., the fastest path) or \( k \) paths with the top-\( k \) least travel costs (e.g., the top-3 fastest paths). Although such services are useful, users may still benefit from richer information, such as the travel time distributions of the returned paths, which may help them make better travel plans.

Consider an example where a user is going to the user’s home \( s \) and an airport \( d \) and there are three paths, \( P_1, P_2, \) and \( P_3 \) connecting \( s \) and \( d \). The travel cost distributions of the three paths are different during time period [8 a.m., 9 a.m.) and [9 a.m., 11 a.m.).

![Figure 4.1: Travel Time Cost Distributions of \( P_1, P_2, \) and \( P_3 \)](image-url)
4.2. PRELIMINARIES

First, $P_2$ and $P_3$ are time-dependent as they vary as shown in Fig. 4.1(a) and (b). Next, consider a scenario where the user needs to leave home at 9 a.m., and the cumulative probability of travel time costs of $P_1$, $P_2$, and $P_3$ are as shown in Fig. 4.1(b). If the user aims to reach the airport within 20 minutes, $P_1$ is preferable because it gives highest probability of doing so; nevertheless, if the user aims to reach the airport within 40 minutes $P_2$ is preferable because it gives the highest probability of reaching the destination when the travel time goes beyond 20 minutes. Thus, we can return $P_1$ and $P_2$ so that users can choose according to their preferences. However, $P_3$ is not returned as it does not yield the highest probability for any case.

This example illustrates how users may have different needs and how travel cost distributions may help users to choose the most appropriate routes according to their needs.

We propose a novel path planning problem, called time-dependent non-dominated stochastic path planning, that is able to return a set of routes that includes the most appropriate routes (e.g., $P_1$ and $P_2$ in Fig. 4.1(b)), but excludes routes that cannot be the most useful in any scenario.

The paper makes four contributions. First, it formalizes the time-dependent non-dominated stochastic path planning problem. Second, it proposes the time-dependent uncertain weight modeling of a road network, which is able to better capture real traffic conditions. Third, it develops a query processing method based on time-dependent uncertain contraction hierarchies to efficiently solve the non-dominated stochastic path planning problem. Fourth, it reports on comprehensive experiments that involve a substantial GPS data set. These offer insight into the efficiency, accuracy, and scalability of the proposed methods.

The remainder of the paper is organized as follows. Section 4.2 covers preliminaries, formulates the problem considered, and proposes methods for building road network with uncertain edges weights. Sections 4.3 proposes an algorithm based on uncertain contraction hierarchies that is able to compute the non-dominated stochastic path planning queries. Section 4.4 reports on the experimental results. Section 4.5 reviews related work, and Section 4.6 concludes.

4.2 Preliminaries

We proceed to cover definitions of important concepts and to formalize the problem.

4.2.1 Road Networks, Paths, and Trajectories

Definition 1 (Time-Dependent Random Variable) The time of a day $TS$ is partitioned into $M$ time periods, denoted as

$$TS = \{tp_1, \ldots, tp_M\}.$$ 

Thus, a time-dependent random variable is a vector of random variables

$$\langle rv_1, \ldots, rv_M \rangle,$$
where \( rv_i \) corresponds to \( tp_i \) for \( i \in [1, M] \).

**Definition 2 (Road Network)** A Road Network is a weighted, directed graph \( G = (V, E, W) \), where \( V \) is a vertex set, \( E \subseteq V \times V \) is an edge set, and \( W \) is an edge weight function. A vertex \( v_i \in V \) represents a road intersection or a road end. An edge \( e_k = (v_i, v_j) \in E \) models a directed road segment that enables travel from vertex \( v_i \) to vertex \( v_j \), and \( e_k.s \) and \( e_k.d \) denote the source and destination of edge \( e_k \) respectively.

Function \( W \) maps an edge to its time-dependent weight. Specifically, in **deterministic weight modeling**, function \( W \) is defined as
\[
W : E \rightarrow \mathbb{R}^+,
\]
where \( \mathbb{R}^+ \) is the set of positive real numbers so that each edge is mapped to a positive real number that indicates the travel cost of traversing the edge. In contrast, in **time-dependent uncertain weight modeling**, function \( W \) is defined as
\[
W : E \rightarrow \mathbb{RV},
\]
where \( \mathbb{RV} \) is a set of time-dependent random variables so that each edge is mapped to a time-dependent random variable that describes the distribution of the travel costs of traversing the edge at different times.

**Definition 3 (Path and Pre-Path)** A path \( P = \langle e_1, e_2, \ldots, e_A \rangle \), where \( A \geq 1 \), is a sequence of edges that connect a sequence of distinct vertices where \( e_i \in E \) and consecutive edges share a vertex (\( e_i.d \) is the same as \( e_{i+1}.s \), \( 1 \leq i < A \)); and the vertices \( e_1.s, e_2.s, \ldots, e_A.s, \) and \( e_A.d \) are distinct. A pre-path of path \( P \), denoted as \( P^{(k)} = \langle e_1, \ldots, e_k \rangle \), consists of the first \( k \) edges in \( P \), where \( 1 \leq k \leq A \).

**Definition 4 (Trajectory)** A Trajectory \( tr = \langle p_1, p_2, \ldots, p_x \rangle \) is a sequence of GPS records. Each GPS record \( r_i = (loc, t) \) specifies a location \( p_i.loc \) (typically with latitude and longitude coordinates) and a timestamp \( r_i.t \). Further, the GPS records in a trajectory are ordered increasingly based on their timestamps, i.e., \( r_i.t < r_{i+1}.t, i \in [1, x] \).

**Definition 5 (Traversal Record)** Given a trajectory set and a road network in which the trajectories occurred, each GPS record in the trajectory can be mapped to a location on an edge in the road network using a map-matching algorithm [67]. After map-matching, each edge is associated with a set of traversal records.

A traversal record is of the form \( tr = (e, t_s, c) \). A record \( tr \) indicates that a traversal occurred on edge \( e \) that started at timestamp \( t_s \) and has travel cost (e.g., travel time or fuel consumption) \( c \).

In particular, a travel time cost can be obtained as the difference between the timestamps of the last and the first GPS records on the edge. Using appropriate vehicular environmental impact models [39, 40], a fuel consumption cost can be computed based on speeds and accelerations, which can be obtained or derived from the GPS records that are mapped to the edge.
4.2.2 Time-dependent Uncertain Edge Weights

Our analysis suggests that the travel cost distributions of an edge generally vary during a day. Take an edge in Denmark as an example. As shown in Fig. 4.2, its travel time distribution during peak hour differs from its off-peak hour travel time distribution.

We therefore partition the time space $TS$ of a day into $M$ time periods of interest, denoted as $TS = \{tp_1, tp_2, \ldots, tp_M\}$, where the uncertain weights of an edge vary significantly in adjacent time periods. Thus, each edge $e$’s time-dependent uncertain weight can be modeled as a time-dependent random variable, and a different random variable then represents $e$’s uncertain edge weight for each time period in $TS$.

Given a set of all historical traversal records on edge $e$ during a time period $tp_i$, it is possible to derive a random variable $X_i$ that describes the distribution of the travel costs associated with traversing edge $e$. This random variable $X_i$ is used as the uncertain weight of edge $e$ in $tp_i$. The time-dependent uncertain weight of an edge in $G$ is denoted as $W(e)$, and for a time $t$, is denoted as $W(e)[j]$, where $t$ is in time period $tp_j$, which is the $j$-th time period in $TS$. Section 4.2.2.1 describes how to compute and represent uncertain edge weights.

4.2.2.1 Uncertain Edge Weights Representation

Let $tp$ be a time period of interest. The cost values observed in historical traversal records on an edge $e$ during $tp$ can be treated as a multiset of cost values $C$, the range of the cost values $R(C)$ is the set of non-duplicated values that occur in $C$. The data distribution of the cost values in $C$ is given by $D(C) = \{(val, prob)\}$, where $val$ indicates a value in $R(C)$ and $prob$ is the number of occurrences of the value in $C$ divided by the total number of values in $C$.

An uncertain edge weight is used to capture the data distribution of travel costs of an edge during a specific time period. A simple yet effective solution is to model
the data distribution as a discrete random variable, and histograms are chosen to represent the travel costs distributions.

In particular, a histogram partitions the range of cost values $R(C)$ into $B$ buckets $\langle b_1, \ldots, b_B \rangle$, where a bucket $b_i = [f_i, l_i]$ indicates a range of the cost values in $C$, where $f_i$ and $l_i$ indicate the starting and ending values of the range.

A histogram $H = \langle (b_1, p_1), \ldots, (b_n, p_n) \rangle$ is a vector of (bucket, prob) pairs, and each bucket $b_i$ is associated with a value $p_i$ that records the probability of the cost values that are in the range indicated by $b_i$, and $\sum_{i=1}^n p_i = 1$. The width of a bucket is defined as $|b_i| = l_i - f_i$. We also use the notation $H_{b_i}$ to represent the $i$-th bucket in $H$ and $H_{p_i}$ to represent $H_{b_i}$’s probability. An equi-width histogram groups contiguous ranges of values into buckets with the same width, i.e., $|b_1| = \ldots = |b_n|$. For a value $x$, $H(x)$ is the probability of $x$ derived from $H$’s data distribution.

We use an equi-width histogram $H$ to represents the uncertain edge weight of an edge $e$, and we let $H_{Min}$ and $H_{Max}$ be the minimum and maximum values in edge $e$’s historical travel costs. The data range $[H_{Min}, H_{Max}]$ is then partitioned into $B$ buckets with equal width

$$\frac{|H_{Max} - H_{Min}|}{B},$$

where $B$ is a tunable parameter that defines the number of buckets.

For example, Fig. 4.3(a) shows the travel times on an edge $e$, and Fig. 4.3(b) shows $e$’s uncertain edge weight represented by an equi-width histogram with 3 buckets. When the number of buckets $B$ varies, different histograms are obtained for the same traversal records data set.

![Figure 4.3: Uncertain Edge Weight of An Edge](image)

It is often the case that not all edges in a road network are covered by the available historical traversal records. For an edge $e$ with no traversal records, we use a histogram $H(e)$ with a single bucket:

$$H(e) = \langle ([w(e), w(e)], 1) \rangle,$$
4.2. PRELIMINARIES

where

\[ w(e) = \frac{\text{len}(e)}{\text{sl}(e)}, \]

where \( \text{len}(e) \) is the length of \( e \) and \( \text{sl}(e) \) is the speed limit of \( e \).

Furthermore, based on a recent study [39] on how to obtain GHG emissions distributions on edges in a road network from GPS records, it follows that histograms can also be used to represent other travel cost types such as GHG emissions. In this paper, we focus on travel time costs, but the proposed techniques can also be applied to GHG emissions costs.

To sum up, to generate the time-dependent uncertain edge weight for an edge \( e \) in a road network \( G \), where time is divided into \( N \) time periods of interest, namely \( \{ t_{p_1}, t_{p_2}, \ldots, t_{p_N} \} \), we compute \( e \)'s uncertain edge weight for each time period of interest.

4.2.3 Uncertain Edge Weight Convolution

Intuitively, the travel cost of a path is the sum of the uncertain edge weights of all the edge it contains; thus, it is uncertain as well. Consider a path \( P = \langle e_1, e_2, \ldots, e_{Q-1}, e_Q \rangle \) that consists of \( Q \) edges. The uncertain travel cost of \( P \) when traversing \( P \) starting in a given time period \( t_p \), denoted as \( W_{t_p}(P) \), can also be represented as a random variable, which is conceptually the sum of the random variables of all edges. Assuming that the uncertain weights of different edges are independent of each other, the sum of uncertain weights can be computed using convolution. Given that a traversal of \( P \) starts and ends during \( t_p \), by sequentially convoluting the uncertain weights during \( t_p \) of the edges in \( P \), we obtain the uncertain travel cost of \( P \).

\[ W_{t_p}(P) = \bigoplus_{i=1}^{Q} W_{t_p}(e_i), \quad (4.1) \]

where \( W_{t_p}(e_i) \) is the uncertain edge weight during \( t_p \) of the \( i \)-th edge in \( P \), and \( \bigoplus \) denotes the convolution of two random variables.

Specifically, given two independent discrete random variables \( X \) and \( Y \) with data ranges \( D(x) \) and \( D(y) \), the convolution of \( X \) and \( Y \), denoted as \( Z = X \oplus Y \), can be computed as follows.

\[ Z(z) = \sum f_X(x) \cdot f_Y(z-x), \forall x \in D(X), \exists z-x \in D(Y). \]

Thus, the probability of random variable \( Z \) having a value \( z \) is the sum of \( f_X(x) \) (the probability of \( X \) having value \( x \)) multiplied by \( f_Y(z-x) \) (the probability of \( Y \) having value \( z-x \)), for all possible \( x \) in \( D(X) \).

To illustrate, let \( P = \langle e_1, e_2 \rangle \) be a path, and let the travel time histograms of \( e_1 \) and \( e_2 \) in time period \( t_p = [8 \text{ a.m., 10 a.m.}] \) be \( H_1 \) and \( H_2 \) as shown in Fig. 4.4(a). Let a trip \( tr_1 \) on \( P \) start at 8:10 a.m., so that the latest time the traversal exits \( P \) is 8:18 a.m. Then, the travel cost of \( P \) is the convolution of \( H_1 \) and \( H_2 \) as shown in Fig. 4.4(b).
CHAPTER 4. A PRACTICAL APPROACH TO ROUTING WITH TIME-VARYING, UNCERTAIN EDGE WEIGHTS

However, if the travel time histograms of $e_1$ and $e_2$ vary in $t_{p1} = [8\ a.m.,\ 9\ a.m.)$ and $t_{p2} = [9\ a.m.,\ 10\ a.m.)$, i.e., as shown in Fig. 4.5 and a trip starts at 8:58 a.m., then the arrival time at $e_2$ overlaps with two time periods that have different travel cost distributions on $e_2$, as the possible arrival time range at $e$ is [8:59 a.m., 9:03 a.m.).

An arrival confidence $ac$ associated with an edge $e$ in $P$ and a time period $t_{p}$ is used to present the confidence of a traversal arriving at edge $e$ during $t_{p}$.

For example, if a trip $tr_2$ traversing $P$ starts at 8:57 a.m., the probability of traversing $e_1$ within 3 minutes is 0.5 and the probability of using more than 3 minutes to traverse $e_1$ is also 0.5 (see Fig. 4.5(a)). Therefore, the arrival confidences at $e_2$ in [8 a.m., 9 a.m.) and in [9 a.m., 10 a.m.) are 0.5 and 0.5 respectively.

In this case, to compute the travel time cost histogram of traversing $P$, we first compute the convoluted histograms $H_1' = H_1 \oplus H_2$ and $H_2' = H_1 \oplus H_4$, which
represent the travel costs of $P$ when arriving at $e_2$ in [8 a.m., 9 a.m.) and in [9 a.m., 10 a.m.), respectively. **Histogram merging** \cite{57} can be employed to merge $H'_1$ and $H'_2$ into the final histogram $\overline{H}$ while taking $H'_1$’s and $H'_2$’s arrival confidence into account. In particular, histogram merging first transforms two histograms $H_i$ and $H_j$ into the corresponding isomorphic histograms $H'_i$ and $H'_j$, where $H'_i$ and $H'_j$ have the same lower bound and upper bound, as well as the same number of buckets $B$. $H'_i$ and $H'_j$ are then merged into a single histogram $\overline{H}$ according to the confidences of these histograms, $ac_i$ and $ac_j$. In particular, $\overline{H}$ is computed using Equation 4.2.

$$\overline{H}.p_k = H'_1.p_k \cdot ac_i + H'_2.p_k \cdot ac_j, \ \forall \ k \in [1, B]$$ (4.2)

### 4.2.4 Problem Definition

**Stochastic dominance** is used to compare two random variables $X$ and $Y$. $X$ stochastically dominates $Y$ if

$$\forall a \in \mathbb{R}^+(CDF_X(a) \geq CDF_Y(a)) \land$$

$$\exists b \in \mathbb{R}^+(CDF_X(b) > CDF_Y(b)),$$

where $CDF_X$ ($CDF_Y$) is the cumulative distribution function (CDF) of random variable $X$ ($Y$).

Since a CDF describes the probability that a random variable $X$ with a given probability distribution will be found at a value less than or equal to $x$, the CDF of the travel cost of path $P$ gives the probability of traversing $P$ with a travel cost that does not exceed $a$. For example the CDF $CDF_X(a)$ of a continuous random variable $X$ can be expressed as the integral of its probability density function $f_X(a)$ as follows.

$$CDF_X(a) = \int_{-\infty}^{a} f_X(a)da$$

When histograms are used to capture probability distributions, the distribution of probability inside a histogram bucket is assumed to be uniform. Thus, the PDF $f_X$ derived from a histogram $H$ is denoted as

$$f_X(x) = \frac{H_{p_k}}{H_{b_k}}, x \in [H_{f_i}, H_{l_i}).$$

A histograms can be transformed into a corresponding CDF as follows.

$$CDF_X(a) = \sum_{x=0}^{a} f_X(x)$$

Figure 4.6 gives an example of a histogram and its corresponding CDF representation.

Given the histograms of paths $P_1$ and $P_2$ with travel cost distribution CDFs represented as piece-wise linear functions $F_1$ and $F_2$, **travel cost dominance** checking is performed using $F_1$ and $F_2$. 
CHAPTER 4. A PRACTICAL APPROACH TO ROUTING WITH TIME-VARYING, UNCERTAIN EDGE WEIGHTS

Definition 6 (Travel Cost Dominance) \( F_1 \) stochastically dominates \( F_2 \) if and only if \( \mathbb{R}^+ \) can be partitioned into contiguous \( N \) sub-domains \( D_1, D_2, \ldots, D_{N-1}, D_N \) such that

\[
\forall i \in [1, N] (x \in D_i \Rightarrow F_1(x) \geq F_2(x)) \land \\
\exists i \in [1, N] (x \in D_i \Rightarrow F_1(x) > F_2(x)).
\]

Definition 7 (Time-dependent Non-dominated Stochastic Path Planning) A Time-dependent Non-dominated Stochastic Path (TNSP) query takes as input source and destination vertices \( s \) and \( d \) and a start time \( t \), and it returns a set of paths from \( s \) to \( d \) that are not stochastically dominated by any other paths starting at time \( t \).

4.2.5 Framework Overview

Figure 4.7 gives an overview of the framework. The system has an offline component and an online component. The offline component takes as input a trajectory set and a road network, and it computes the uncertain edge weights for the road network. The online component provides a real-time service that computes non-dominated stochastic path queries according to users’ requests.

4.3 Query Processing

We proceed to propose efficient query processing algorithms to answer time-dependent non-dominated stochastic path queries.

4.3.1 Baseline Method

With uncertain edge weights modeled as random variables, given source and destination vertices \( s \) and \( d \) together with the starting time \( t \), time-dependent non-dominated
stochastic path queries can be computed by our baseline method in Algorithm 7. In particular, the baseline method is based on Dijkstra’s algorithm [30]. In Algorithm 7, $TNSP$ is the set of time-dependent non-dominated stochastic paths, and $Q$ is a priority queue of partially explored paths, where the partially explored paths are ordered by their minimum possible travel cost. Thus, the path that potentially yields the least possible travel cost is explored first. Starting from the source vertex $s$, Algorithm 7 iteratively expands the path $p$ at the front of $Q$ to each neighbor $n$ of $p$’s last vertex $v$. Once $d$ is reached, $P$ is added to $TNSP$ if it is not dominated by a path in $TNSP$. Algorithm 7 terminates when $Q$ is empty.

To be more specific, Algorithm 7 starts from the source vertex $s$, thus $(s, h)$ is added to $Q$ (line 2), where $h$ is an empty histogram. For each iteration (lines 4–9), a pre-path $p$ is expanded to each neighbor vertex $n$ of $p$’s last vertex $v$, as long as $n$ is not already included in $P$. Algorithm 8 (line 9) is called to prune all explored and partially-explored paths by performing travel cost dominance checking, and to determine whether an expanded path $p$ should be further expanded. When a path $p$ being explored reaches the destination $d$ and it is not dominated by a path in $TNSP$, it is then added to the candidate path set $TNSP$.

$TNSPUpdate$, described as Algorithm 8, it is a helper function that efficiently prunes the search space. The hashmap $PRE$ maps a visited vertex $v$ to all explored pre-paths that reach $v$. For a path $p$, $h$ is the travel cost distribution of $p$. It first determines the minimum and maximum travel costs ($t_{\text{min}}$ and $t_{\text{max}}$) to reach $v$ via pre-path $p$, and it then expands $p$ by appending $e$ to $p$. The travel cost histogram of the expanded $p$ is computed in (line 5–12). If $t_{\text{min}}$ and $t_{\text{max}}$ fall into the same time period of interest $tp$, the estimated travel cost of the expanded $p$ is the convolution of $h$ and the travel cost histogram of $e$ in $tp$. However, when the $t_{\text{min}}$ and $t_{\text{max}}$ fall into different time periods $tp_1$ and $tp_2$, $[t_{\text{min}}, t_{\text{min}})$ overlaps with the time range of $tp_1$ and $tp_2$. In Algorithm 8, if $p$ reaches the destination $d$, $DominanceCheck$ is called to perform travel cost dominance checking against all the explored candidate paths in $TNSP$ with $p$ (line 15); otherwise, dominance checking is performed against all
Algorithm 7: TNSPBaseline

Input:
Source vertex: $s$; Destination vertex: $d$;
Starting time: $t$;

Output:
Non-dominated Stochastic Path: $TNSP$;

1: $h \leftarrow$ empty histogram;
2: $Q$.enqueue($((s), h)$);
3: while $Q$ is not empty do
4:   $(p, h) \leftarrow Q$.dequeue();
5:   $v \leftarrow p$’s last vertex;
6:   for all $n \in v$’s neighbors and $n \notin p$ do
7:     Edge $e \leftarrow (v, n)$;
8:     if $e \in E$ then
9:       TNSPUpdate($p, e, h, t$);
10: Return $TNSP$;

the pre-path of $v$ in $PRE[v]$, where $v$ is the last vertex of $p$ (line 17). Algorithm 9 checks dominance between a path $p$ and each path $p’$ in a path set $S$. According to Lemma 1, if $p$ is dominated by a path $p’ \in S$, $p$ can be safely eliminated; on the other hand, if $p$ dominates a path $p’ \in S$, $p’$ can be eliminated from $S$.

As shown in Lemma 1, suppose path $p$ is being considered and that the last vertex in $p$ is $v$. If there exists another path $p’$ between $s$ and $d$ that dominates $p$, paths starting with pre-path $p$ are not worth exploring; on the other hand, if $p$ dominates $p’$, all explored paths starting with $p’$ should be eliminated.

Lemma 1 Let two path $P_1$ and $P_2$ share the same ending vertex $v$, and let $e$ be an edge starting at $v$, $P'_1 = P_1 + e$, and $P'_2 = P_2 + e$. If travel cost distribution of $P_1$ stochastically dominates that of $P_2$ in a time period $t_p$, the travel cost distribution of $P'_1$ stochastically dominates that of $P'_2$ in $t_p$.

Proof 1 Let $H_1$, $H_2$, and $H_e$ be the travel cost histograms of $P_1$, $P_2$, and $e$. Therefore, $H'_1 = H_1 \oplus H_e$, and $H'_2 = H_2 \oplus H_e$. If $P'_1$ does not stochastically dominate $P'_2$, there exists a value $x$ such that $CDF_{H'_1}(x) < CDF_{H'_2}(x)$. Thus,

$$\sum_{y=0}^{x} H'_1(y) < \sum_{y=0}^{x} H'_2(y),$$

which indicates

$$\sum_{y=0}^{x} \left(\sum_{z=0}^{y} H_1(z) \cdot H_e(y - z)\right) < \sum_{y=0}^{x} \left(\sum_{z=0}^{y} H_2(z) \cdot H_e(y - z)\right),$$
where \( z \) is all possible values in \([0, y]\) for \( y \in [0, x] \). According to Equation 4.3, this is equivalent to

\[
\sum_{y=0}^{x} \left( \sum_{z=0}^{y} H_1(z) \cdot H_e(x-y) \right) < \sum_{y=0}^{x} \left( \sum_{z=0}^{y} H_2(z) \cdot H_e(x-y) \right).
\]

However, if \( H_1 \) stochastically dominates \( H_2 \), which guarantees \( \text{CDF}_{H_1}(x) \geq \text{CDF}_{H_2}(x) \),

\[
\sum_{z=0}^{y} H_1(z) \geq \sum_{z=0}^{y} H_2(z).
\]

This yields that

\[
\sum_{y=0}^{x} \left( \sum_{z=0}^{y} H_1(z) \cdot H_e(x-y) \right) \geq \sum_{y=0}^{x} \left( \sum_{z=0}^{y} H_2(z) \cdot H_e(x-y) \right),
\]

which contradicts the above inequality. Therefore, \( P'_1 \) stochastically dominates \( P'_2 \).

In particular, for two travel cost histograms \( H_1 \) and \( H_2 \), for a value \( x \),

\[
\sum_{y=0}^{x} \left( \sum_{z=0}^{y} H_1(z) \cdot H_2(y-z) \right)
\]

\[
= \sum_{z=0}^{0} H_1(z) \cdot H_2(0-z) + \ldots + \sum_{z=0}^{x} H_1(z) \cdot H_2(x-z)
\]

\[
= (H_1(0) + H_1(1) + \ldots + H_1(x)) \cdot H_2(0)+
\]

\[
(H_1(0) + H_1(1) + \ldots + H_1(x-1)) \cdot H_2(1)+
\]

\[
\ldots + H_1(0) \cdot H_2(x)
\]

\[
= \sum_{z=0}^{x} H_1(z) \cdot H_2(0) + \sum_{z=0}^{x-1} H_1(z) \cdot H_2(1)+
\]

\[
\ldots + \sum_{z=0}^{0} H_1(z) \cdot H_2(x)
\]

\[
= \sum_{y=0}^{x} \left( \sum_{z=0}^{y} H_1(z) \cdot H_2(x-z) \right)
\]  \hspace{1cm} (4.3)

Take the road network in Fig. 4.8 as an example and assume that a user goes from \( v_2 \) to \( v_3 \) starting at 9:00 a.m. The travel cost histograms of \( e_1-e_4 \) for the corresponding time period \([9 \text{ a.m.}, 11 \text{ a.m.}]\), namely \( t_p \), are shown in Fig. 4.9. The travel costs of the trip do not overlap with several time periods with different travel cost distributions, as the maximum travel time is \( H(e_3)_{\text{max}} + H(e_4)_{\text{max}} = 60 + 60 = 120 \) seconds. Table 4.1 shows the process of computing the TNSP set given source \( v_2 \), destination \( v_3 \), and starting time 9:00 a.m.
Algorithm 8: TNSPUpdate

Input:
Pre-path explored: \( p \); Edge to explore: \( e \);
Starting time: \( t \); \( p \)'s travel cost histogram: \( h \);

1: \( t_{min} \leftarrow t + h_{min} \);
2: \( t_{max} \leftarrow t + h_{max} \);
3: \( p \).append\( (e) \);
4: \( v \leftarrow \) \( p \)'s last vertex;
5: if \( t_{min} \) and \( t_{max} \) fall into the same time period \( tp \) then
6: \( h' \leftarrow h \oplus \mathcal{H}_{tp}(e) \);
7: else
8: \( c_1 \leftarrow \) confidence of reaching \( e \) in \( tp_1 \);
9: \( c_2 \leftarrow \) confidence of reaching \( e \) in \( tp_2 \);
10: \( h_1 \leftarrow h \oplus \mathcal{H}_{tp_1}(e) \);
11: \( h_2 \leftarrow h \oplus \mathcal{H}_{tp_2}(e) \);
12: \( h' \leftarrow \text{merge}(c_1 \cdot h_1, c_2 \cdot h_2) \);
13: \( dom \leftarrow \) False;
14: if \( v == d \) then
15: \( dom \leftarrow \text{DominanceCheck}(h', \text{NSP}) \);
16: else
17: \( dom \leftarrow \text{DominanceCheck}(h', \text{PRE}[v]) \);
18: if \( dom == \) False then
19: if \( v == d \) then
20: \( \text{NSP}.\text{insert}(p) \);
21: else
22: \( Q.\text{enqueue}((p, h')) \);
23: \( \text{PRE}[v].\text{insert}(p) \);

4.3.2 Contraction Hierarchies

Although Algorithm 7 can solve the time-dependent non-dominated stochastic path planning problem, the performance deteriorates quickly as longer paths are considered. We propose a time-dependent uncertain contraction hierarchy based method to improve the query performance for large road networks, where contraction hierarchies (CH) used in [38] is a technique that speeds up shortest path planning for road networks with deterministic weights.

For a road network \( G = (V, E) \), where \( V \) and \( E \) are vertex and edge sets in \( G \). CH employs a bijective function \( \phi : V \rightarrow \{0, \ldots, |V| - 1\} \) to assign each vertex an importance value. We use the subscript of a vertex \( v \) to indicate the its importance, i.e., vertex \( v \) is the least important vertex if \( \phi(v) \) is 0 and if \( i < j \), \( \phi(v_i) < \phi(v_j) \).

CH contracts the vertices according to the increasing order of vertices’ importance values. Contracting a vertex \( v_0 \) means removing \( v_0 \) from the original \( G \) without changing the shortest path distances between the remaining (more important)
Algorithm 9: DominanceCheck

Input:
The travel cost histogram of a path: $h$;  
Pre-path set: $S$;

Output:

True: if $h$ is dominated by a path in $S$;  
False: if $h$ is not dominated by any path in $S$;

1: $\text{dom} \leftarrow \text{False}$;  
2: for all Path $p \in S$ do  
3: if $H(p)$ dominates $h$ then  
4: $\text{dom} \leftarrow \text{True}$;  
5: Break;  
6: else if $H(p)$ is dominated by $h$ then  
7: $S$.remove($p$);

![Figure 4.8: Example Road Network](image)

![Figure 4.9: Travel time histograms for time period $tp$](image)
### Table 4.1: Running Example

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q.enqueue((v_2, h))$</td>
<td>initialize $Q$</td>
</tr>
<tr>
<td>$(v_2, h) \leftarrow Q.dequeue()$</td>
<td>pop from $Q$</td>
</tr>
<tr>
<td>TNSPUpdate($p, e_1, h, t$);</td>
<td>evaluate $e_1$</td>
</tr>
<tr>
<td>$p.append(e_1)$</td>
<td>$p = (e_1)$</td>
</tr>
<tr>
<td>$h' \leftarrow h \oplus H_{tp}(e_1)$</td>
<td>$h$ is $p$’s histogram</td>
</tr>
<tr>
<td>$Q.enqueue((p, h')) &amp; PRE[v].insert(p)$</td>
<td>$p$ is not dominated</td>
</tr>
<tr>
<td>TNSPUpdate($p, e_3, h, t$);</td>
<td>evaluate $e_3$</td>
</tr>
<tr>
<td>$p.append(e_3)$</td>
<td>$p = (e_3)$</td>
</tr>
<tr>
<td>$h' \leftarrow h \oplus H_{tp}(e_1)$</td>
<td>$h$ is $p$’s histogram</td>
</tr>
<tr>
<td>$Q.enqueue((p, h')) &amp; PRE[v].insert(p)$</td>
<td>$p$ is not dominated</td>
</tr>
<tr>
<td>$(v_2, v_0) \leftarrow Q.dequeue()$</td>
<td>pop from $Q$</td>
</tr>
<tr>
<td>TNSPUpdate($p, e_2, h, t$);</td>
<td>evaluate $e_2$</td>
</tr>
<tr>
<td>$p.append(e_2)$</td>
<td>$p = (e_1, e_2)$</td>
</tr>
<tr>
<td>$h' \leftarrow h \oplus H_{tp}(e_1)$</td>
<td>$h$ is $p$’s histogram</td>
</tr>
<tr>
<td>$TNSP.insert(p)$</td>
<td>$p$ is not dominated</td>
</tr>
<tr>
<td>$(v_2, v_1) \leftarrow Q.dequeue()$</td>
<td>pop from $Q$</td>
</tr>
<tr>
<td>TNSPUpdate($p, e_4, h, t$);</td>
<td>evaluate $e_4$</td>
</tr>
<tr>
<td>$p.append(e_4)$</td>
<td>$p = (e_3, e_4)$</td>
</tr>
<tr>
<td>$h' \leftarrow h \oplus H_{tp}(e_4)$</td>
<td>$h$ is $p$’s histogram</td>
</tr>
<tr>
<td>$p$ is not added to $TNSP$</td>
<td>$p$ is dominated by $\langle e_1, e_2 \rangle$</td>
</tr>
<tr>
<td>return $TNSP$</td>
<td>$TNSP$ set is returned</td>
</tr>
</tbody>
</table>
4.3. QUERY PROCESSING

Taking the road network shown in Fig. 4.8 as an example, assume that vertex \( v_0 \) is the vertex assigned with the least import vertex order number and is to be contracted. If the edge weights of the road network are as shown in Table 4.2 after contracting \( v_0 \), \( e_1 \) and \( e_2 \) are removed, and the shortcut edge \( \langle v_2, v_3 \rangle \) that represents \( e_1 \) and \( e_2 \) as a single edge (shown as a dashed line in Fig. 4.10) is inserted with a weight that equals to the sum of the weights of \( e_1 \) and \( e_2 \) because \( P = \langle e_1, e_2 \rangle \) is the lowest-cost path between vertices \( v_2 \) and \( v_3 \). In addition, the original edges that a shortcut edge represents can be used at query processing time.

In contrast, if a witness path is found, a shortcut edge does not need to be added. A witness path is a path that has an equivalent or lower cost than the path being considered for contraction. Thus, \( v_0 \) can be removed without adding a shortcut edge to represent \( \langle e_1, e_2 \rangle \) if its cost is not smaller than that of \( \langle e_3, e_4 \rangle \). For instance, if the edge weights are as shown in Table 4.3, shortcut edge \( \langle v_2, v_3 \rangle \) is not needed because path \( \langle e_3, e_4 \rangle \) is now a witness path between \( v_2 \) and \( v_3 \).

![Figure 4.10: Example Road Network (v0 contracted)](image)

<table>
<thead>
<tr>
<th>Path</th>
<th>Travel Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \langle e_1 \rangle )</td>
<td>2</td>
</tr>
<tr>
<td>( \langle e_2 \rangle )</td>
<td>5</td>
</tr>
<tr>
<td>( \langle e_3 \rangle )</td>
<td>5</td>
</tr>
<tr>
<td>( \langle e_4 \rangle )</td>
<td>4</td>
</tr>
<tr>
<td>( \langle e_1, e_2 \rangle )</td>
<td>7</td>
</tr>
<tr>
<td>( \langle e_3, e_4 \rangle )</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 4.2: Travel Costs Setting 1

<table>
<thead>
<tr>
<th>Path</th>
<th>Travel Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \langle e_1 \rangle )</td>
<td>4</td>
</tr>
<tr>
<td>( \langle e_2 \rangle )</td>
<td>6</td>
</tr>
<tr>
<td>( \langle e_3 \rangle )</td>
<td>5</td>
</tr>
<tr>
<td>( \langle e_4 \rangle )</td>
<td>2</td>
</tr>
<tr>
<td>( \langle e_1, e_2 \rangle )</td>
<td>10</td>
</tr>
<tr>
<td>( \langle e_3, e_4 \rangle )</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 4.3: Travel Costs Setting 2

After contracting all the vertices, all the shortcut edges are added to the original road network, which yields a contracted road network.
4.3.3 Time-dependent Uncertain Contraction Hierarchy

To build a time-dependent contraction hierarchy where time is partitioned into \( N \) time periods of interest, we first generate a uniform node order using the default parameters of a Contraction Hierarchy \([38]\). Then we construct uncertain contraction hierarchies for each time period of interest separately, taking the uncertain edge weight for each time period into account. Therefore, shortcut edges are generated for all time periods, and the shortcut edge set \( E^+ = E_1^+ \cup \ldots \cup E_N^+ \) represents the union of shortcut edges added for contraction hierarchies of all time periods of interest.

Recall the road network \( G \) shown in Fig. 4.8, where the vertex ordering is given as \( \phi(v_0) < \phi(v_1) < \phi(v_2) < \phi(v_3) \). Let \( P_1 = \langle e_1, e_2 \rangle \) and \( P_2 = \langle e_3, e_4 \rangle \). The only possible shortcut to consider in \( G \) is \( e_s = \langle v_2, v_3 \rangle \). Assume that time is divided into 3 time periods of interest, \( t_{p1}, t_{p2}, \) and that the \( t_{p3}, \) and edges have different uncertain travel costs in these time periods, as shown in Figs. 4.11–4.13. Then the convoluted uncertain travel costs of \( P_1 \) and \( P_2 \) vary in the different time periods as shown in Figs. 4.14–4.16. We decide whether to add \( e_s \) for each time period.

![Travel time histogram for time period \( t_{p1} \)](image)

Figure 4.11: Travel time histogram for time period \( t_{p1} \)

For time period \( t_{p1} \), the travel costs of \( P_1 \) and \( P_2 \) do not dominate each other; thus, no shortcut needs to be added in this case.

As shown in Fig. 4.15, the travel cost of \( P_1 \) dominates that of \( P_2 \); thus, shortcut edge \( e_s \) is added, and the travel cost histogram is represented as that of \( P_1 \).

Moreover, for time period \( t_{p3} \), the travel cost of \( P_1 \) is dominated by that of \( P_2 \). As a result, shortcut edge \( e_s \) is added; however, it represents \( P_2 \), and its travel cost distribution is represented using that of \( P_2 \).

Consequently, different shortcut edges may be added for different time periods, and the travel cost of a shortcut between two vertices may vary in different time periods.

Given an uncertain road network \( G = (V, E) \), Algorithm \([10]\) describes how uncertain contraction is done for a road network with respect to the given vertex order.
4.3. QUERY PROCESSING

Figure 4.12: Travel time histograms for time period $tp_2$

Figure 4.13: Travel time histograms for time period $tp_3$

Figure 4.14: Path $P_1$ and $P_2$ do not dominate each other in time period $tp_1$
(v_1, v_2, \ldots, v_{n-1}, v_n). The algorithm contracts the vertices sequentially by invoking Algorithm 11 for each vertex. The shortcut edges are added to shortcut edge set $E_{\text{shortcut}}$ in each iteration, and the resulting uncertain road network can be denoted as $G = (V, E \cup E_{\text{shortcut}})$.

Algorithm 11 describes the process of contracting a vertex $v$ in road network $G = (V, E)$. For each vertex $v$ in $G$, the incoming neighbor set $V_{\text{in}}(v)$ is the set of vertices where $\forall v' \in V_{\text{in}}(v) (\langle v', v \rangle \in E)$. Similarly, the outgoing neighbor set $V_{\text{out}}(v)$ is the set of vertices where $\forall v' \in V_{\text{out}}(v) (\langle v, v' \rangle \in E)$. During the contraction process, we use edge set $E_{\text{dom}}$ to store all the paths that are dominated by another explored path between the same source and destination. Paths in $E_{\text{dom}}$ can be skipped when they are considered. Moreover, edge set $E_{\text{ndom}}$ is used to store all the paths that cannot
be ignored when they are considered. They are used to check whether there are paths dominated by them. Nevertheless, a path $P = \langle s, \ldots, x \rangle \in E_{dom}$, it is added to $E_{shortcut}$, as there exists another path $P'$ between $s$ and $x$ such that $P$ and $P'$ cannot dominate each other.

**Algorithm 10: ContractRoadNetwork**

**Input:**
A Road Network $G$;  
Vertex order $\langle v_1, v_2, \ldots, v_{n-1}, v_n \rangle$;
1: $i \leftarrow 1$;
2: while $i \leq n$ do
3:  ContractVertex($v_i$);
4:  $i \leftarrow i + 1$;

**Algorithm 11: ContractVertex**

**Input:**
A vertex $v$ in Road Network $G$;
1: for all $v_i \in v$‘s incoming neighbor set $v_{in}$ do
2:  for all $v_o \in v$‘s outgoing neighbor set $v_{out}$ do
3:    path $P \leftarrow \langle v_i, v, v_o \rangle$;
4:    witnessFound $\leftarrow$ FALSE;
5:    if $P \in$ edge set $E_{dom}$ then
6:      continue;
7:      for all paths $P'$ between $v_i$ and $v_o$ && $|P'| \leq$ HopLimit do
8:        if $P$ dominates $P'$ then
9:          if $|P'| == 2$ then
10:             Add $P'$ to dominated edge set $E_{dom}$;
11:         else
12:           if $P$ and $P'$ do not dominate each other then
13:             if $|P'| == 2$ then
14:               Add $P'$ to dominated edge set $E_{ndom}$;
15:               witnessFound $\leftarrow$ TRUE
16:           if witnessFound $==$ FALSE && $P \notin E_{ndom}$ then
17:             Add $\langle v_i, v, v_o \rangle$ as a shortcut edge to $E_{shortcut}$;
18: Delete all edges connected to $v$;

There are three cases for the dominance checking between two paths $P_1$ and $P_2$ as shown in Figs. 4.14–4.16, i.e., $P_1$ and $P_2$ do not dominate each other, $P_1$ dominates $P_2$, and $P_1$ is dominated by $P_2$.

To illustrate, for time period $t_{p1}$, we first contract vertex $v_0$ and consider the path $P_1 = \langle v_2, v_0, v_3 \rangle$. Another path $P_2 = \langle v_2, v_1, v_3 \rangle$ exists between $v_2$ and $v_3$, and $P_1$ and $P_2$ do not dominate each other. Therefore, no shortcut edge is added for $P_1$, and
CHAPTER 4. A PRACTICAL APPROACH TO ROUTING WITH
TIME-VARYING, UNCERTAIN EDGE WEIGHTS

$P_2$ is added to $E_{ndom}$. We only need to remove edges connected to $v_0$ to compute the contraction for $v_0$. We then proceed to contract vertex $v_1$, the only path contains $v_1$ is $P_2 = \langle v_0, v_1, v_3 \rangle$ is already added to $E_{ndom}$, thus it is not added as a shortcut edge. After edges connected to $v_1$ are removed, it is then contracted. As there are no more edges remaining, we contract vertices $v_2$ and $v_3$ immediately.

Similarly, for time period $t_{p_2}$, we first start from vertex $v_0$ and consider the path $P_1 = \langle v_2, v_0, v_3 \rangle$, and there is another path $P_2 = \langle v_2, v_1, v_3 \rangle$ found between $v_2$ and $v_3$ and $P_1$ dominates $P_2$ and there is no path can dominate $P_1$, $P_1$ is therefore added as a shortcut edge and $P_2$ is added to dominated edge set $E_{dom}$. We then remove edges connected to $v_0$, then contraction for $v_0$ is done. We then proceed to contract vertex $v_1$. The only path that contains $v_1$ is $P_2 = \langle v_2, v_1, v_3 \rangle$ that has already been added to $E_{dom}$. Thus, no shortcut edge is added for it. After edges connected to $v_1$ are removed, we contract vertices $v_2$ and $v_3$ directly.

Finally, for time period $t_{p_3}$, we first start from vertex $v_0$ and consider the path $P_1 = \langle v_2, v_0, v_3 \rangle$. Another path $P_2 = \langle v_2, v_1, v_3 \rangle$ is found between $v_2$ and $v_3$, and $P_1$ is dominated by $P_2$. Therefore no shortcut edge is added for $P_1$. We then remove edges connected to $v_0$, and contraction for $v_0$ is done. We then proceed to contract vertex $v_1$. The only path contains that $v_1$ is $P_2 = \langle v_2, v_1, v_3 \rangle$ and it is not dominated by any path between $v_2$ and $v_3$. Thus, a shortcut edge is added. Then all edges connected to $v_1$ are removed, and we finally contract vertices $v_2$ and $v_3$ without adding additional shortcut edges.

4.3.4 CH-Based Method

Let a road network $G_{orig} = (V, E_{orig})$ be given and assume that time is divided into $N$ time periods of interest such as peak and off-peak periods. Then $G_{orig}$’s time-dependent uncertain contraction hierarchy is represented as $G = (V, E)$, where $E = E_{orig} \cup E^+$, $E^+ = E^+_1 \cup \ldots \cup E^+_N$, and $E^+_i$ contains the shortcut edges added for $i$-th time period. Thus, given a source and destination pair $(s, d)$ in a time-dependent uncertain contraction hierarchy $G$ and a start time $t$, our algorithm finds all non-dominated stochastic paths in $G$. As in Algorithm 7, we use a priority queue $Q$ to store all the pre-paths from $s$ to a vertex $v$ in $G$ together with the corresponding travel cost histograms. The minimum travel costs of pre-paths are used as the key in $Q$. Again, a hashmap $PRE$ is used to keep the mapping of a vertex $v$ to all explored pre-paths $\langle s, \ldots, v \rangle$ that reach $v$, such that the pre-paths to a vertex $v$ in $PRE[v]$ are not dominated by any other explored pre-path from $s$ to $v$ in $PRE[v]$. During the process of path exploration, if a pre-path $p$ from $s$ to $v$ is dominated by another explored pre-path $p'$ from $s$ to $v$, $p$ is eliminated on the fly.

Algorithm 12 describes how to find all time-dependent non-dominated stochastic paths between $s$ and $d$ starting at time $t$. Initially, for each time period $t_{p_i}$, we do a backward Dijkstra search from $d$ in $G_{\downarrow}$ to find edge set $E'_{t_i}$ that contains edges that can reach $d$ in $t_{p_i}$ (lines 1–3). The query is performed on graph $G' = (V, E_{\uparrow} \cup E')$, where $E' = E'_1 \cup \ldots \cup E'_N$. Algorithm 13 (line 12) is called to prune all explored paths and pre-paths by performing travel cost dominance check, and to determine whether
4.3. QUERY PROCESSING

an expanded path $p$ should be further expanded. Once a path $p$ being explored reaches the destination $d$, it is added to the path set $TNSP$ if it is not dominated by a path in $TNSP$.

**Algorithm 12: TNSPCH**

**Input:**
- Source vertex: $s$;
- Destination vertex: $d$;
- Starting time: $t$;

**Output:**
- Non-dominated Stochastic Path: $TNSP$;

1: for all $i \in [1, N]$ do
2: Backward Dijkstra search from $d$ in $G$ ↓;
3: $E' ← E' \cup E_i'$;
4: $h ←$ empty histogram;
5: $Q$.enqueue($((s), h)$);
6: while $Q$ is not empty do
7: $(p, h) ← Q$.dequeue();
8: $v ← p$’s last vertex;
9: for all $n ∈ v$’s neighbors and $n \not∈ p$ do
10: Edge $e ← (v, n)$;
11: if $e ∈ E ↑ \cup E'$ then
12: TNSPCHUpdate($p, e, h, t$);
13: Return $TNSP$;

Algorithm 13 describes the process of exploring an edge $e$ in $G$ with respect to the minimum and maximum travel costs of reaching $e$. As the parameter $h$ represents the travel cost distribution as a histogram, $h$ also indicates the minimum and maximum travel costs ($t_{min}$ and $t_{max}$) to reach $e$ through pre-path $p$ (lines 1–2). Path $p$ is then extended by appending edge $e$ to it. To compute the travel cost histogram of $p$, two cases must be considered. First, if $[t_{min}, t_{max})$ falls into a single time period $tp$ so that traversing $e$ yields a unique travel cost histogram $H_{tp}(e)$ (lines 7–10), the resulting travel cost histogram is the convolution of $h$ and $H_{tp}(e)$. Second, when $t_{min}$ and $t_{max}$ belong to two different time periods $tp_1$ and $tp_2$ that yield different travel costs to traverse $e$, namely $h_1$ and $h_2$, the distribution of $h$ is used to determine the confidences $c_1$ and $c_2$ of reaching $e$ in $tp_1$ and $tp_2$ (lines 12–13). Thus, $h_1$ and $h_2$ are convoluted with $h$ separately. Then, based on $c_1$ and $c_2$, they are merged into a final histogram to represent the travel cost histogram of path $p$.

To illustrate how Algorithm 12 works, consider road network shown in Fig. 4.17. Shortcut edge $e_s$ was added between $v_2$ and $v_3$ for time periods $tp_1$ and $tp_2$. Given a source $s$ and destination $d$, a starting time $t$, and the travel cost histogram $h$ of pre-path $p = ⟨p, ..., v_2⟩$. The possible time range to reach $v_2$ can be denoted as $T_{v_2} = [t_{min}, t_{max})$ which overlaps with $tp_1$ and $tp_2$. In particular, the two edges represented by a shortcut edge $e_s = e_1 + e_2$ can be used at query time. Thus, if $e_s$
Algorithm 13: TNSPCHUpdate

Input:
Pre-path explored: $p$, Edge to explore: $e$;
Starting time: $t$, $p$’s Travel cost histogram: $h$;

1: $t_{\text{min}} \leftarrow t + h_{\text{min}}$;
2: $t_{\text{max}} \leftarrow t + h_{\text{max}}$;
3: if $e$ represents different shortcuts in time periods covered by $[t_{\text{min}}, t_{\text{max}})$ then
   4: \text{return};
5: $p$.append($e$);
6: $v \leftarrow p$’s last vertex;
7: if $t_{\text{min}}$ and $t_{\text{max}}$ fall into the same time period $tp$ then
   8: $h' \leftarrow h \oplus H_{tp}(e)$;
9: else
10: $c_1 \leftarrow$ confidence of reaching $e$ in $tp_1$;
11: $c_2 \leftarrow$ confidence of reaching $e$ in $tp_2$;
12: $h_1 \leftarrow h \oplus H_{tp_1}(e)$;
13: $h_2 \leftarrow h \oplus H_{tp_2}(e)$;
14: $h' \leftarrow \text{merge}(c_1 \cdot h_1, c_2 \cdot h_2)$;
15: $\text{dom} \leftarrow \text{False}$;
16: if $v = d$ then
   17: $\text{dom} \leftarrow \text{DominanceCheck}(h', \text{NSP})$;
18: else
19: $\text{dom} \leftarrow \text{DominanceCheck}(h', \text{PRE}[v])$;
20: if $\text{dom} = \text{False}$ then
   21: if $v = d$ then
      22: \text{NSP}.insert($p$);
   23: else
      24: \text{Q}.enqueue(($p, h'$));
   25: \text{PRE}[v] . insert($p$);
4.4. EMPIRICAL STUDY

represents a unique shortcut edge in \( tp_1 \) and \( tp_2 \), the confidences of arriving at \( v_2 \)
in \( tp_1 \) and \( tp_2 \) are can be computed as \( c_1 \) and \( c_2 \) using \( h \). Accordingly, \( h_1 \) and \( h_2 \)represent the travel cost histograms of \( e_s \) in \( tp_1 \) and \( tp_2 \). The possible time range toreach \( v_3 \), \( T_{v_3} \), is given by Equation \( 4.4 \):

\[
T_{v_3} = \left[ t_{min} + \min(e_{s,tp_1},min,e_{s,tp_2},min), \right. \\
\left. t_{max} + \max(e_{s,tp_1},max,e_{s,tp_2},max) \right]
\]

(4.4)

Here, \( e_{s,tp_1},min \) represents the minimum travel cost of \( e_s \) in time period \( tp_1 \). The
time-dependent Routing Example

4.3.5 Speedup Techniques

Stall-on-demand is a technique that is used to prune the forward search from \( s \) and the backward search from \( t \) in CH. It can be adapted for use in time-dependent uncertain CH. In particular, when considering a path \( P = \langle s, \ldots, v \rangle \) and its last vertex is \( v \), suppose there exists a previously explored path \( P' = \langle s, \ldots, v_x \rangle \) and there exists an edge \( e = \langle v_x, v \rangle \) and \( \phi(v_x) > \phi(v) \). Then, if the travel cost histogram of \( P \) is stochastically dominated by that of path \( P' + e \), further exploration from \( P \) can be stalled as \( P \) cannot lead to a non-dominated stochastic path in the final result.

Take the search between source and destination vertices \( s \) and \( d \) in Fig. 4.18 as an example. When we consider path \( P_2 = \langle s, e_1, e_3, e_4 \rangle \) whose last vertex is \( v_4 \), there exists path \( P_1 = \langle s, e_2, e_5 \rangle \) and an edge \( e_6 = \langle v_5, v_4 \rangle \). With \( P'_1 = P_1 + e_6 \), if histogram \( H_{P'_1} \) can stochastically dominate \( H_{P_2} \), we can stall the further search from \( P_2 \) and can prune this path.

4.4 Empirical Study

In this section, we evaluate our methods using a road network of Denmark and a substantial GPS data set associated with it.
CHAPTER 4. A PRACTICAL APPROACH TO ROUTING WITH TIME-VARYING, UNCERTAIN EDGE WEIGHTS

4.4.1 Experimental Settings

Road Networks. To evaluate our methods that compute time-dependent non-stochastic path queries, we use the road networks of Aalborg (AA), North Jutland (NJ), and Jutland (JU) from OpenStreetMap[^1] where Aalborg is the largest city in North Jutland, and North Jutland is one of the regions in Jutland. Thus, AA is part of NJ, and NJ is part of JU. The sizes of the 3 road networks are described in Table 4.4. To get the best accuracy of map-matching, we extract road segments from OpenStreetMap with the finest granularity.

<table>
<thead>
<tr>
<th>Name</th>
<th>Number of Vertices</th>
<th>Number of Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>5,192</td>
<td>13,032</td>
</tr>
<tr>
<td>NJ</td>
<td>69,838</td>
<td>164,456</td>
</tr>
<tr>
<td>JU</td>
<td>680,414</td>
<td>1,680,524</td>
</tr>
</tbody>
</table>

GPS Records. We use more than 180 million GPS records collected at 1 HZ (i.e., one GPS record per second) in Denmark during in 2007 and 2008 to generate the uncertain edge weights in the road network. An existing map matching tool [67] is used to match the GPS records to the road networks.

Method Evaluations. We randomly generate 1,000 source-destination pairs \((u, v)\) and starting times \(t\) for query in order to evaluate our methods. The source-destination pairs are grouped by their Euclidean distances. The query sets are shown in Table 4.5. We report the average query processing time and TNSP count in Section 4.4.3.1 and Section 4.4.4.1. The time of a day is partitioned into 5 time periods, \([7 \text{ a.m., } 9 \text{ a.m.}]\) and \([3 \text{ p.m., } 5 \text{ p.m.}]\) are peak hours, and \([0 \text{ a.m., } 7 \text{ a.m.}], [9 \text{ a.m., } 3 \text{ p.m.}], \) and \([5 \text{ p.m., } 12 \text{ a.m.}]\) are off-peak hours.

[^1]: http://www.openstreetmap.org/
4.4. EMPIRICAL STUDY

Table 4.5: Distance Range of Source-Destination Pairs

<table>
<thead>
<tr>
<th>Query Set</th>
<th>Distance Range (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>[0,2), [2, 5), [5, 10), [10, 20)</td>
</tr>
<tr>
<td>NJ</td>
<td>[0,10), [10, 20), [20, 50), [50, 100)</td>
</tr>
<tr>
<td>JU</td>
<td>[0,50), [50, 100), [100, 200), [200, 300)</td>
</tr>
</tbody>
</table>

**Implementation.** All the preprocessing and query processing algorithms are implemented in C++. A machine running Debian GNU/Linux with a 16-core Intel Xeon 2GHz CPU, 32GB main memory, and 4TB external memory is used for all experiments.

**Preprocessing.** For CH-based method, we build contraction hierarchies for each time period of interest separately for all the road networks. For a single time period, the average preprocessing times for AA, NJ, and JU are 20 seconds, 40 seconds, and 101 seconds, respectively.

**4.4.2 Concrete Query Example**

Figure 4.19 shows an example of the result of a time-dependent non-dominated stochastic path query between A and B in Aalborg, Denmark. The blue path $P_1$ and the red path $P_2$ are returned as they do not stochastically dominate each other in travel cost distributions. Their travel time distributions are shown in Fig. 4.20.

This example shows that non-dominated stochastic path between two vertices in a road network can be significantly different, indicating that our proposal suggests meaningful path options to the users.
CHAPTER 4. A PRACTICAL APPROACH TO ROUTING WITH TIME-VARYING, UNCERTAIN EDGE WEIGHTS

4.4.3 Query Performance

4.4.3.1 Distance Between Sources and Destinations

We first study the impact of distance between source-destination pairs on the query processing time. Figure 4.21 shows the query performance of the baseline and CH-based methods while varying the distance between the source and destination vertices. Figure 4.21(a) shows the performance of Algorithm 7 (baseline method) on Road Network AA. The results indicate that the query processing time increases immensely when the distance between the source and destination grows, which makes the search space grow exponentially, so it is not possible to employ Algorithm 7 on large road network. Figure 4.21(b) shows the performance of Algorithm 12 (CH-based method). Here, the query processing time does not increase significantly as the source-destination distance increase; thus, it is feasible to employ Algorithm 12 for large road networks. The following study focuses on experiments with the baseline method on road network AA road network and with the CH-based method on road network JU.
4.4. EMPIRICAL STUDY

In particular, Fig. 4.22 shows the running time of the CH-based method on the three road networks. It shows the average query time for each source-destination query group. The results suggest that the CH-based method can achieve good query performance by searching the vertices in ascending order of each vertex’s order in the road network within a limited search space. Thus, queries on a road network at national scale (i.e., the JU network) can be handled.

4.4.3.2 Peak vs. Off-Peak Hours

We consider peak and off-peak hours separately, to observe the impact in the running times of our algorithms of starting times in different time periods. Figure 4.23 shows the query performance of each source-destination group in road network JU. The results suggest that the overall query processing time during peak hours is slightly longer than that during off-peak hour. This is likely due to more edges having uncertain weights during peak hours.
CHAPTER 4. A PRACTICAL APPROACH TO ROUTING WITH
TIME-VARYING, UNCERTAIN EDGE WEIGHTS

4.4.3.3 Varying the Histogram Bucket Count

Intuitively, query performance may vary when the number of buckets used in the
histograms is varied, i.e., a finer histogram granularity better estimates travel cost
distributions at the cost of a longer query process time. Figure 4.24(a) shows the
impact of changing bucket count in histograms on query performance for the baseline
using road network AA, and Figure 4.24(b) shows the performance changes while
varying the histogram bucket number for CH-based methods using all three road
networks. The results indicate that using more buckets in histograms leads to longer
query process time. This is because the histogram convolution time grows.

![Histogram Bucket Count](image1)

Figure 4.24: Varying the Histogram Bucket Count

4.4.3.4 Varying the Uncertain Weight Percentage

Similar to Section 4.4.3.3, to study how the percentage of edges with uncertain edge
weight affects query performance, we generate uncertain edge weights for those
dges that do not have uncertain edge weights. The synthetic edge weights fol-
low a normal distribution and are computed using edge lengths and speed limits.
Figure 4.25 shows the performance when varying the percentage of edges having
uncertain weights in different road networks. The query process time grows as more
edges have uncertain edge weights in each road network.

4.4.4 TNSP Count

4.4.4.1 Distance Between Sources and Destinations

We first study the average number of time-dependent non-dominated stochastic paths
and the impact of source-destination distance in Fig. 4.26 for the three road network.
As the distance between source and destination vertices increases, the number of
TNSP also increases, while it does not increase sharply as travel distance grows.
4.4. EMPIRICAL STUDY

4.4.4.2 Peak vs. Off-Peak Hours

To observe the impact of starting time in peak versus off-peak periods on the number of TNSP paths returned, we conduct experiments with source-destination pairs by varying the starting times in peak hours and off-peak hours. Figure 4.27(a) shows the average TNSP path number when starting time is not categorized, and Fig. 4.27(b) shows the different average number of TNSP paths returned for peak and off-peak hours. The results suggest that the average number of TNSP paths for queries starting in peak hours is larger than that for off-peak hour. This is mainly due to more edges being covered by historical GPS data in peak hours, meaning that more edges have uncertain weight during peak hours.

4.4.4.3 Varying the Histogram Bucket Count

Next, we study how the number of histogram buckets impacts the number of TNSP paths. More buckets used in histograms provide more detailed information of the historical travel costs. Figure 4.28 indicates that the number of returned TNSP paths...
grows as the number of buckets in a histogram increases. However, the number of returned paths only grows moderately.

### 4.4.4.4 Varying the Uncertain Weight Percentage

Further, we study how the number of TNSP paths change when varying the percentage of the edges that have uncertain edge weights. Figure 4.29 shows the change of the number of returned TNSP paths when the percentage of edges having uncertain weights changes. It suggests that when more edges have uncertain edge weights, more TNSP paths are returned while the number of returned paths does not increase rapidly.
4.5 Related Work

Finding the shortest path between a source and a destination in a road network with deterministic edge weights has been a popular research topic for several decades. The classic algorithm for solving the problem is Dijkstra’s algorithm [30]. The performance of Dijkstra’s algorithm deteriorates quickly as the number of edges grows, which makes it unattractive for routing in large road networks.

Recently, indexing and pre-processing techniques have been proposed that speed up shortest path computation. There are two main categories of state-of-the-art techniques [86]. In the first category, based on the spatial coherence property of the road network, all shortest paths are compressed and can be used to efficiently answer queries [74,76]. Alternatively, the second category ranks the vertices in the road network by their importance [38]. For example, vertices located at major highway conjunctions have higher importance than those located at country road conjunctions. Query algorithms then search the vertices in the order of their importance. However, these speed-up techniques only work for road networks with deterministic weights, not for road networks with uncertain weights. In this paper, we extend contraction hierarchies to support uncertain weights.

A few studies consider road network with uncertain weights in different application scenarios [27, 43, 53, 59, 64]. However, we note that these proposals lack a detailed description of how to obtain the uncertain weights and that most of the existing studies rely on randomly generated synthetic weights. In contrast, we employ a substantial GPS data set to derive uncertain weights that reflect the real traffic conditions. Next, one study [43] employs a best-first search method for path selection in uncertain road networks. Different heuristic functions are used to identify the best path to explore, and thresholds are required [43]. In contrast, our method only takes source and destination and a start time as input. To the best of our knowledge, there is only previous work on small road networks with synthetic uncertain edge weights, while our proposed method returns all the non-dominated stochastic paths on a road network at nation size with uncertain edge weights derived from real traffic data.

4.6 Conclusion

We first describe how to model time-dependent travel cost uncertainty in a road network. Based on our uncertain road network representation, we formalize the problem of find non-dominated stochastic paths, which helps users to make better travel plans by providing the travel cost distribution of the paths. We also propose baseline and advanced algorithms for this problem. Finally, experiments are conducted to gain insight into the efficiency and performance of our algorithms.

In future work, it is of interest to take more travel cost types (e.g., GHG emissions, travel distance) into account to enable non-dominated multi-cost stochastic path queries. In addition, the current algorithms can be extended to support personalized non-dominated stochastic shortest paths.
Chapter 5

Towards Personalized, Context-Aware Routing

Abstract

A driver’s choice of a route to a destination may depend on the route’s length and travel time, but a multitude of other, possibly hard-to-formalize aspects may also factor into the driver’s decision. There is evidence that a driver’s choice of route is context dependent, e.g., varies across time, and that route choice also varies from driver to driver. In contrast, conventional routing services support little in the way of context dependence, and they deliver the same routes to all drivers.

We study how to identify context-aware driving preferences for individual drivers from historical trajectories, and thus how to provide foundations for personalized navigation, but also professional driver education and traffic planning. We provide techniques that are able to capture time-dependent and uncertain properties of dynamic travel costs, such as travel time and fuel consumption, from trajectories, and we provide techniques capable of capturing the driving behaviors of different drivers in terms of multiple dynamic travel costs. Further, we propose techniques that are able to identify a driver’s contexts and then to identify driving preferences for each context using historical trajectories from the driver. Empirical studies with a large trajectory data set offer insight into the design properties of the proposed techniques and suggest that they are effective.

5.1 Introduction

Travel in road networks is an important aspect of our lives, and a variety of navigation services exist that offer suggested routes when supplied with a source, a destination, and an optional departure time. Such services provide all users with the same routes, and they do not take into account a user’s context beyond possibly the user’s departure time. Specifically, navigation services often recommend shortest routes or
fastest routes, where the travel times are derived from speed limits \[11\] rather than from actual driving conditions, e.g., peak vs. off-peak traffic.\footnote{A recent study \cite{92} from Microsoft Research Asia suggests that although some services (e.g., Google Maps, Bing Maps, and Yahoo! Maps) can display time-dependent traffic conditions, such information is not used in their routing services.}

A recent study documents how the routes provided by a major navigation service often fail to agree with the routes chosen by local drivers, who may have detailed knowledge of, and experience with, local driving conditions \cite{26}. In addition, different drivers, and even the same driver, often do not follow the same route between the same source and destination.

To illustrate how different drivers can have different preferences, Fig. 5.1 shows two routes used by two different drivers. The thin (blue) route is longest, but also fastest, as it involves largely highway driving. The bold (red) route is shorter, and it may yield a lower fuel consumption.

These observations suggest that a personalized and context aware routing service has the potential to deliver routes that better match the preferences of drivers than do existing routing services. In particular, better routing may be achieved by the modeling of driver preferences and more thorough modeling of traffic characteristics, as summarized next.

**Multiple Criteria:** When drivers choose routes, they may consider more than one criterion, e.g., travel distance, travel time, fuel consumption, toll cost, number of
traffic lights. Routes that only consider one criterion, e.g., shortest routes or fastest routes, may not fully fulfill drivers’ needs.

**Time-Dependent Uncertainty:** Dynamic criteria such as travel time and fuel consumption are time dependent. For example, traversing a road during peak hours may take much longer than that during off-peak hours. Moreover, time dependent costs are generally uncertain. For instance, aggressive driving may consume more fuel than moderate driving, but may also reduce travel time. The uncertainty may also vary across time. For example, the uncertainty may be high during off-peak hours because drivers can better drive as fast or slow as they want; during peak hours, congestion may force drivers to drive similarly, thus reducing uncertainty.

**Context-Aware Driving Preferences:** Preferences generally vary across drivers. For example, some drivers may prefer fastest routes, while other drivers may choose routes that represent trade-offs between travel time and distance. A single driver’s preferences may also depend on the context. For instance, a driver may only care about travel time during peak hours, while caring also about fuel consumption and toll costs during a weekend trip.

Thus, a routing service should support more than one criterion, should capture the time-dependent uncertainty of dynamic criteria, and should incorporate context-aware driving preferences. To this end, we model multiple criteria as functions, where each function concerns a travel cost, e.g., travel distance or time, and maps each road-network edge to that cost. We propose techniques that are able to obtain travel costs that are time-dependent and uncertain based on a collection of trajectories. We then provide techniques capable of modeling a driver’s behavior in terms of such travel costs. The resulting driver behavior models are then used to reduce uncertainty for individual drivers, and to identify different contexts for individual drivers. For each driver, and in each identified context, we learn a driving preference vector with the help of personalized skyline routes.

Identifying driving preferences are not only useful for providing better, personalized and context-aware navigation services, but also for other purposes. For instance, fleet managers, e.g., FlexDanmark, are interested in aiding their drivers in making more eco-friendly driving decisions while also considering travel times.

To the best of our knowledge, this is the first proposal of general techniques that are able to effectively identify context-aware driving preferences from large trajectory data while considering multiple, time-dependent, and uncertain travel costs. Specifically, the paper makes four contributions. First, it proposes techniques that are able to derive time-dependent, uncertain travel costs from trajectories. Second, it proposes techniques that enable identification of a driver’s contexts. Third, it presents techniques for automatically generating training data using personalized skyline routes and learning context-aware driving preference based on the training data. Fourth, it reports on a comprehensive empirical evaluation in a realistic setting with a substantial GPS trajectory data set, and it offers insight into the design properties of the proposed techniques.

---

2FlexDanmark is a large fleet manager in Denmark. See [https://www.flexdanmark.dk/](https://www.flexdanmark.dk/)
CHAPTER 5. TOWARDS PERSONALIZED, CONTEXT-AWARE ROUTING

The paper is organized as follows. Section 5.2 covers the problem setting. Section 5.3 describes how to derive different travel costs from trajectory data. Section 5.4 details context identification and preference learning algorithms. Section 5.5 reports on the empirical evaluation. Section 5.6 covers related work. Finally, conclusions and research directions are offered in Section 5.7.

5.2 Problem Setting and Definition

We use blackboard bold upper case letters to denote sets, e.g., $E$; bold letters for column vectors, e.g., $w$; and calligraphic letters for sequences, e.g., $R$. An overview of key notation is provided in Table 5.1.

A road network is modeled as a directed, labeled graph $G = (V, E)$, where $V$ and $E \subseteq V \times V$ is a vertex set and an edge set, respectively. A vertex $v_i \in V$ models a road intersection or an end of a road. An edge $e_k = (v_i, v_j) \in E$ models a directed road segment, indicating that travel is possible from its source $v_i$ to its destination $v_j$. We use the notation $e_k.s$ and $e_k.d$ to denote the source and destination of edge $e_k$.

A route $R = \langle r_1, r_2, \ldots, r_A \rangle$, where $A \geq 1$, is a sequence of edges that connect a sequence of distinct vertices, where $r_i \in E$ and consecutive edges share a vertex ($r_i.d$ is the same as $r_{i+1}.s$, $1 \leq i < A$); and the vertices $r_1.s$, $r_2.s$, $\ldots$, $r_A.s$, and $r_A.d$ are distinct. The first $a$ edges in route $R$ constitute a pre-route of route $R$, denoted as $R^{(a)} = \langle r_1, r_2, \ldots, r_a \rangle$, where $1 \leq a \leq A$.

A trajectory $T = \langle p_1, p_2, \ldots, p_B \rangle$ is a sequence of GPS records pertaining to a trip, where each record $p_i$ specifies a (location, time) pair of a vehicle, where $p_i.time < p_j.time$ if $1 \leq i < j \leq B$. A trajectory is also associated with a driver identifier, denoted as $DID(T)$, indicating who made the trajectory.

Map matching [67] is used to map a GPS record to a specific location on an edge in the road network. Map matching transforms a trajectory $T$ into a sequence of edge records $\langle l_1, l_2, \ldots, l_C \rangle$. A record $l_i$ is of the form $(e, t, GPS, d)$, where $e$ is an edge traversed by trajectory $T$; $t$ is the time when the traversal of edge $e$ starts; $GPS = \langle p_j, p_{j+1}, \ldots, p_k \rangle$ contains the GPS records mapped to edge $e$; and $d$ indicates the identifier of the driver who made the traversal. The sequence of the edges in the sequence of edge records is referred as the route of trajectory $T$, denoted as $R_T = \langle l_1.e, l_2.e, \ldots, l_C.e \rangle$.

With these definitions in place, we can formulate the problem addressed: Given a set of trajectories $\mathcal{T}_R$, each associated with a driver identifier, and a set of travel costs of interest (e.g., travel distance, travel time, fuel consumption), context-aware driving preference learning aims to identify distinct contexts for each driver and aims to identify a preference vector $w$ for each context of each driver. Preference vector $w$ specifies the relative importance on each travel cost of interest in its context.

For instance, a driver may prefer the fastest route in the morning in order to avoid being late, but may consider to save fuel consumptions on other occasions. Context-aware driving preference learning based on the driver’s historical trajectories should identify the two different contexts and the corresponding preference vectors. The
### 5.2. PROBLEM SETTING AND DEFINITION

Table 5.1: Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G, \forall, \mathcal{E}$</td>
<td>Road network, vertex set, edge set.</td>
</tr>
<tr>
<td>$\mathcal{R}, \mathcal{R}^{(a)}$</td>
<td>A route, a pre-route with the first $a$ edges.</td>
</tr>
<tr>
<td>$\mathcal{T}, \mathcal{R}_{\mathcal{T}}$</td>
<td>A trajectory, the route of trajectory $\mathcal{T}$.</td>
</tr>
<tr>
<td>$l_{i}$</td>
<td>An edge record.</td>
</tr>
<tr>
<td>$SC^{(i)}(e_{k})$</td>
<td>The $i$-th static cost of edge $e_{k}$.</td>
</tr>
<tr>
<td>$DC^{(j)}(e_{k})$</td>
<td>The $j$-th dynamic cost of edge $e_{k}$.</td>
</tr>
<tr>
<td>$SCR^{(i)}(\mathcal{R})$</td>
<td>The $i$-th static cost of route $\mathcal{R}$.</td>
</tr>
<tr>
<td>$DCR^{(j)}(\mathcal{R})$</td>
<td>The $j$-th dynamic cost of route $\mathcal{R}$.</td>
</tr>
<tr>
<td>$d_{m}, \mathcal{T}\mathcal{R}<em>{d</em>{m}}$</td>
<td>A driver, the set of trajectories from driver $d_{m}$.</td>
</tr>
<tr>
<td>$\alpha, MLT$</td>
<td>Finest interval of interest, mean lifetime.</td>
</tr>
<tr>
<td>$N$</td>
<td>Total number of travel costs of interest.</td>
</tr>
<tr>
<td>$ER(\mathcal{T})$</td>
<td>Efficiency ratio vector of trajectory $\mathcal{T}$.</td>
</tr>
<tr>
<td>$\tilde{ER}(\mathcal{T})$</td>
<td>Ordinal efficiency ratio vector of trajectory $\mathcal{T}$.</td>
</tr>
<tr>
<td>$PR_{d_{m}}^{(j)}(\mathcal{T})$</td>
<td>Personal ratio of trajectory $\mathcal{T}$ w.r.t. driver $d_{m}$ and the $j$-th dynamic cost.</td>
</tr>
<tr>
<td>$Cl_{i}$</td>
<td>A context.</td>
</tr>
<tr>
<td>$\mathbf{w}$</td>
<td>Driving preference vector.</td>
</tr>
<tr>
<td>$c(\mathcal{R}, t_{s})$</td>
<td>Cost vector of using route $\mathcal{R}$ at time $t_{s}$.</td>
</tr>
<tr>
<td>$\mathcal{T}_{ord}$</td>
<td>The set of trajectories whose ordinal efficiency ratios are $ord$.</td>
</tr>
</tbody>
</table>
contexts in this example are defined in terms of temporal aspects, e.g., morning vs. non-morning. Each context has a preference vector—the preference vector has a larger weight for travel time in the morning context and a larger weight for fuel consumption in the non-morning context.

When a driver plans a trip, a routing module identifies an appropriate context and uses the corresponding preference vector to find an “optimal” route that minimizes the weighted sum (w.r.t. the preference vector $w$) of the travel costs of interest among all possible routes. An overview of the whole procedure is shown in Fig. 5.2.

5.3 Modeling Travel Costs

We categorize travel costs as static or dynamic, and we provide techniques to obtain accurate dynamic costs from trajectories.

5.3.1 Static vs. Dynamic Travel Costs

When planning a trip from a source to a destination at a trip starting time, a prerequisite for choosing an appropriate route is to know the travel costs of using different routes from the source to the destination at the trip starting time. We categorize travel costs into static costs and dynamic costs, as shown in Table 5.2.

Static costs are represented by deterministic values (e.g., real numbers), and they never change across time. For example, distance is a static cost. Static costs can often be extracted from various digital maps such as OpenStreetMap[^1].

In contrast, dynamic costs are time-dependent and, typically, also uncertain. For example, Fig. 5.3a shows the travel times of trajectories on two edges w.r.t. the starting times of the trajectories. Fig. 5.3a) shows data for an edge with clear morning and afternoon peaks, around 8:00 and 16:00, respectively; and at a particular time, the travel times are not deterministic, but follow some distribution; and the distribution differs across different times. Although the edge covered in Fig. 5.3b) has no clear peak periods, its distribution of travel times also varies across time.

[^1]: [http://www.openstreetmap.org](http://www.openstreetmap.org)
5.3. MODELING TRAVEL COSTS

Table 5.2: Categorization of Travel Costs

<table>
<thead>
<tr>
<th></th>
<th>Static Costs</th>
<th>Dynamic Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Representation</strong></td>
<td>Deterministic values.</td>
<td>Dynamic cost functions</td>
</tr>
<tr>
<td><strong>Derived From</strong></td>
<td>Geometrical and physical properties</td>
<td>GPS trajectories from the road network.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>Distances.</td>
<td>Travel times, fuel consumption.</td>
</tr>
</tbody>
</table>

(a) Two peak periods  
(b) No clear peak periods

Figure 5.3: Examples of Time-Dependent, Uncertain Travel Times

A dynamic cost is modeled as a dynamic cost function $F : T \rightarrow \mathbb{R}^V$, where $T$ is the time domain of interest, e.g., a day or a week, and $\mathbb{R}^V$ is a set of all possible random variables. Specifically, $F$ takes as input a time and returns a random variable that describes the distribution of some travel cost at that time. Dynamic costs are generally not available in digital maps, but can be derived from GPS trajectories, detailed in Section 5.3.2. Based on the travel cost categorization, we proceed to define travel costs of edges and routes, respectively.

5.3.1.1 Travel Costs of Edges

We assume that drivers are interested in $N$ different travel cost types. For each type of cost, we maintain a function that assigns travel costs to all edges. Specifically, for
the $i$-th Static Cost ($SC$) type, function $SC^{(i)} : E \rightarrow \mathbb{R}^+$ is maintained. For example, if the $i$-th cost type is travel distance, and the length of edge $e_k$ is 1.5 km, we have $SC^{(i)}(e_k) = 1.5$.

For the $j$-th Dynamic Cost ($DC$) type, function $DC^{(j)} : E \rightarrow \mathbb{F}$ maps each edge to its $j$-th dynamic cost. Thus, $\mathbb{F}$ is a set of elements with signature $T \rightarrow \mathbb{R}V$. This means that $DC^{(j)}(e_k)$ is a dynamic cost function. When a time $t$ is supplied as argument, a random variable is thus returned. Put differently, $(DC^{(j)}(e_k))(t)$ is a random variable that describes the distribution of the $j$-th dynamic cost at time $t$.

To ease the presentation, we use $SC^{(DI)}$, $DC^{(TT)}$, and $DC^{(FC)}$ to represent the functions assigning travel distance ($DI$), travel time ($TT$), and fuel consumption ($FC$), respectively. We use $X[x]$ to denote the probability function of random variable $X$. For example, if $X$ is a Gaussian random variable with mean $\mu$ and standard deviation $\delta$ as its parameters, then $X[x] = \frac{1}{\sqrt{2\pi}\delta^2} e^{-\frac{(x-\mu)^2}{2\delta^2}}$.

### 5.3.1.2 Travel Costs of Routes

The cost of a route is derived from the costs of the edges in the route. The static cost of a route $R = \langle r_1, r_2, \ldots, r_A \rangle$ is the sum of the static costs of the edges in the route. In particular, the $i$-th Static Cost of Route $R$, $SCR^{(i)}(R)$, is $\sum_{k=1}^{A} SC^{(i)}(r_k)$. For instance, the distance of a route is the sum of the distances of the edges in the route. Next, since dynamic costs are time-dependent, we consider a traversal of route $R$ starting at $t_s$. Since the cost of traversing an edge at a particular time is uncertain, the cost of traversing route $R$ at $t_s$ is also uncertain. In particular, the $j$-th Dynamic Cost of Route $R$ at $t_s$ is denoted as a random variable $DCR^{(j)}(R, t_s)$. Thus, determining the travel cost of traversing route $R$ at $t_s$ is equivalent to deriving the probability function of the random variable, i.e., $DCR^{(j)}(R, t_s)[x]$.

Recall that $R^{(k)}$ denotes the pre-route of $R$ consisting of the first $k$ edges. Since $R^{(1)} = \langle r_1 \rangle$, and the traversal on $r_1$ starts at $t_s$, the probability function of the cost random variable of $R^{(1)}$ is defined by Equation 5.1.

$$DCR^{(j)}(R^{(1)}, t_s)[x] = (DC^{(j)}(r_1))(t_s)[x]$$

When traversing a route, the time when starting to traverse a subsequent edge keeps changing. The starting time for the traversal of edge $r_k$ ($k > 1$) is $t_s$ plus the time needed to traverse pre-route $R^{(k-1)}$. Since the travel time of an edge is uncertain, the travel time of pre-route $R^{(k-1)}$ is also uncertain. Thus, the starting time on edge $r_k$ is uncertain as well.

Based on the above, we denote the cost random variable of edge $r_k$ where the traversal on route $R$ starts at $t_s$ by $DCE^{(j)}(r_k, t_s)$. The probability function of the
random variable of edge \( r_k \) is defined in Equation 5.2.

\[
DCE^{(j)}(r_k, t_s)[x] = \int_0^{+\infty} \left( \int \left( DC^{(j)}(r_k) \right)(t_s + \tau)[x] \right) d\tau
\]

The probability of the cost of \( r_k \) being \( x \) (i.e., Part A) is the integral of the probability of the travel time (TT) of pre-route \( R^{(k-1)} \) being \( \tau \) (i.e., Part B) multiplied by the probability of the cost of traversing \( r_k \) at \( t_s + \tau \) being \( x \) (Part C) over all possible \( \tau \).

Next, the cost random variable of pre-route \( R^{(k)} \) \((k > 1)\) is the sum of the cost random variable of pre-route \( R^{(k-1)} \) and the cost random variable of edge \( r_k \). The probability function of the cost random variable of pre-route \( R^{(k)} \) is defined in Equation 5.3.

\[
DCR^{(j)}(R^{(k)}, t_s)[x] = \int_0^{+\infty} \left( \int \left( DCR^{(j)}(R^{(k-1)}, t_s)[\tau] \right) \right) d\tau
\]

Here, the probability of the cost of pre-route \( R^{(k)} \) being \( x \) (i.e., Part D) is the integral of the probability of the cost of pre-route \( R^{(k-1)} \) being \( \tau \) (i.e., Part E) multiplied by the probability of the cost of \( r_k \) being \( x - \tau \) (i.e., Part F) over all possible \( \tau \).

Based on the above, an algorithm that computes the \( j \)-th dynamic cost random variable of route \( R \) at starting time \( t_s \) is given in Algorithm 14.

**Algorithm 14: RouteDynamicCostComputation**

- **Input**: Route: \( R = (r_1, r_2, \ldots, r_A) \); Time: \( t_s \); CostType: \( j \);
- **Output**: Random Variable: \( DCR^{(j)}(R, t_s) \);

1. Random Variable \( preRouteCost \) ← null;
2. Random Variable \( edgeCost \) ← null;
3. \( preRouteCost \) ← Using Equation 5.1 based on \( t_s \);
4. for \( k = 2; k \leq A; k++ \) do
   5. \( edgeCost \) ← Using Equation 5.2 based on \( preRouteCost \);
   6. \( preRouteCost \) ← Using Equation 5.3 based on \( edgeCost \);
7. return \( preRouteCost \)

### 5.3.2 Instantiating Dynamic Cost Functions

To derive dynamic costs, we need to instantiate dynamic cost functions of the edges in a road network based on trajectories from the road network.
When instantiating the dynamic cost function of an edge $e_k$, we consider the edge records $L_{e_k} = \{l_m|l_m.e = e_k\}$ (recall the definition of edge records in Section 5.2). Different types of dynamic costs can be derived from the GPS records in an edge record, i.e., $l_i.GPS$. For instance, travel times can be derived by getting the difference between the timestamps of the first and last GPS records; and fuel consumptions can be computed based on instantaneous (or average) speeds and accelerations, which all can be obtained or derived from the GPS records using appropriate vehicular environmental impact models [39, 40].

5.3.2.1 Discrete Approach

A discrete approach [88] of instantiating a dynamic cost function is to partition a period of interest (e.g., a week day) into fixed-length intervals. A random variable, which is typically represented by its probability function, can be estimated based on the costs observed in the edge records that occurred in an interval.

This approach is intuitive and easy to be implemented; however, it has two problems. First, the distributions of random variables may change suddenly across interval boundaries, (e.g., jumping from peak to off-peak at a boundary). However, in reality, the peak periods should disappear gradually instead of suddenly. Second, if only a small number of edge records occurred in an interval, the obtained random variable may be over-fitting to these edge records. If no edge records occurred in an interval, the random variable cannot even be estimated.

5.3.2.2 Continuous Approach

To better capture the real, continually changing traffic behavior, we propose a time-decaying based technique to instantiate dynamic cost functions while smoothing the distributions across time.

We partition a day into $D = \lceil \frac{24 \times 60}{\alpha} \rceil$ intervals, where parameter $\alpha$ (minutes) specifies the finest-granularity interval of interest in minutes, e.g., 15 minutes. Consider edge $e_k$ and the $j$-th dynamic cost type. For each interval $I_i$ ($1 \leq i \leq D$), we identify a set of edge records $L_{e_k}^{I_i} = \{l_m|l_m.e = e_k \land l_m.t \in I_i\}$. Based on $L_{e_k}^{I_i}$, we obtain a multiset $C_{e_k,j}^{I_i} = \{(\text{cost}, \text{count})\}$ for the $j$-th dynamic cost type on edge $e_k$, where $\text{cost}$ indicates a cost value (e.g., the travel time on edge $e_k$ from a traversal that occurred in interval $I_i$), and $\text{count}$ indicates how many times the cost value is observed in the interval.

Next, we estimate a random variable for each interval. Since traffic changes gradually, when estimating the random variable in the $i$-th interval, we consider not only the costs in the interval, i.e., $C_{e_k,j}^{I_i}$, but also the costs in its nearby intervals. Intuitively, the costs in the $i$-th interval should be considered as more important than the costs in nearby intervals; and the costs from nearby intervals should be considered as more important than the costs from further away intervals.

We call this the time-decaying property, and we capture it using an exponential decay function $\text{decay}(t) = e^{-\frac{t}{\text{MLT}}}$, where $\text{MLT}$ is the mean lifetime parameter.
5.3. MODELING TRAVEL COSTS

Based on this, the relative importance of the costs in the $x$-th interval w.r.t. the $i$-th interval is given by Equation 5.4.

$$ decay(x, i) = \begin{cases} e^{-\frac{\alpha}{MLT} \cdot |x-i|} & \text{if } |x-i| \leq \frac{D}{2}; \\ e^{-\frac{\alpha}{MLT} \cdot \min(|x+D-i|,|x-D-i|)} & \text{otherwise}; \end{cases} \quad (5.4) $$

For example, we set $\alpha = 15$ and we have $D = 96$ intervals. When estimating the random variable during the 1-st interval [0:00, 0:15), the 2-nd interval [0:15, 0:30) and the 96-th interval [23:45, 0:00) should have the same high importance. According to Equation 5.4, the 2-nd interval’s relative importance is $e^{-\frac{15}{MLT} \cdot |2-1|} = e^{-\frac{15}{MLT}}$, and the 96-th interval’s is $e^{-\frac{15}{MLT} \cdot \min(|96+96-1|,|96-96-1|)} = e^{-\frac{15}{MLT}}$. In Section 5.5.2, we evaluate empirically the effect of varying the $MLT$ parameter.

By using the decay function, we actually consider all the costs on $e_k$ when estimating the random variable on edge $e_k$ for the $i$-th interval. However, the costs from different intervals are assigned different importance. In particular, if we have a record $(\text{cost}, \text{count})$ from the $x$-th interval, we use $(\text{cost}, \text{count} \cdot decay(x, i))$ when estimating the random variable for the $i$-th interval.

Figure 5.4(a) shows the counts of fuel consumption values in an interval on an edge before and after using the decaying function. For example, 14 ml and 15 ml are observed 3 and 4 times in the interval, respectively. Since they are also observed frequently in the nearby intervals, after applying the decaying function, the count of 14 ml and 15 ml increase to 9.34 and 5.47, respectively.

In addition to capturing the continually changing traffic behavior better, use of the time-decaying function gives us two additional advantages over the discrete approach [88]. First, for an interval without any costs, e.g., a midnight interval, we are able to learn a random variable based on the costs from all the remaining intervals. Second, for an interval with limited number of costs, e.g., only 1 or 2 costs, we are able to obtain a more robust random variable which avoids over-fitting to the limited
number of costs observed in the interval. In Section 5.5.2, we study the accuracy of using the discrete and the continuous approaches. Algorithm 15 describes the procedure of instantiating the $j$-th dynamic cost for edge $e_k$.

Algorithm 15: InstantiateDynamicCostFunction

Input: EdgeRecords: $L_{e_k}$; int $j$; Double $\alpha$;
Output: DynamicCostFunction: $DC^{(j)}(e_k)$;
1 $D \leftarrow \lceil \frac{24 \cdot 60}{\alpha} \rceil$;
2 for $i = 1; \; i \leq D; \; i++$ do
3 $\text{MultiSet } C \leftarrow \emptyset$;
4 for $x = 1; \; x \leq D; \; x++$ do
5 for each $(\text{cost}, \text{count})$ in $C_{e_k,j}$ do
6 Add $(\text{cost}, \text{count} \cdot \text{decay}(x, i))$ into $C$;
7 Estimate a random variable $RV$ based on $C$;
8 $DC^{(j)}(e_k)(I_i) \leftarrow RV$;
9 return $DC^{(j)}(e_k)$;

Next, we consider how to learn a random variable from a multiset of travel cost values after applying the decaying function (line 7 in Algorithm 15). Since a random variable is represented by its probability function, we first need to choose an appropriate probability function. We choose to use continuous, parametric probability functions to represent random variables, due to its superiority compared to discrete probability functions, e.g., histograms [88]. In particular, we choose to use Gaussian Mixture Models (GMMs) to represent random variables, where a GMM is a weighted sum of several Gaussian distributions (a.k.a., Gaussian components).

The use of GMMs has two benefits. First, it is able to approximate any complex probability functions [21]. This is important because the distributions of dynamic costs on many edges cannot be modeled well using any single probability function, e.g., a Gaussian or exponential distribution [88]. Second, using GMMs makes Algorithm 14 efficient, because integral operations can be efficiently computed based on the limited number of parameters in GMMs.

Finally, we learn appropriate parameters based on the multiset to identify the GMM. Detailed algorithms are omitted but can be found in existing work [87]. Fig. 5.4(b) shows that a GMM with three Gaussian components is learned to represent the distribution of fuel consumptions, based on the multiset (after applying the decaying function) shown in Fig. 5.4(a).

5.4 Identifying Context-Aware Driving Preferences

We propose techniques to identify a driver’s contexts (Section 5.4.1), and to identify driving preferences for each context of a driver (Section 5.4.2). Finally, we briefly
5.4. IDENTIFYING CONTEXT-AWARE DRIVING PREFERENCES

describe how to integrate the identified context and driving preference into a routing algorithm that enables personalized, context-aware navigation (Section 5.4.3).

5.4.1 Identifying Contexts

In different contexts, a driver has different driving preferences. We first introduce a concept called efficiency ratio (in Section 5.4.1.1) and then propose three different strategies to model drivers’ driving behavior and to compute efficiency ratios (in Section 5.4.1.2). By clustering the efficiency ratios derived from a driver’s trajectories, we are able to identify a driver’s contexts (in Section 5.4.1.3).

5.4.1.1 Efficiency Ratios

Each trajectory made by driver $d_m$ provides meaningful information about the driver’s driving preferences. Let driver $d_m$ have trajectory set $\mathcal{T}_{d_m} = \{ T | \text{DID}(T) = d_m \}$. Consider the $j$-th cost type. The $j$-th cost of trajectory $T \in \mathcal{T}_{d_m}$, denoted as $T.c^{(j)}$, is the sum of the $j$-th costs of the edges in the route used by $T$, i.e., $\mathcal{R}_T$. For example, the distance (or travel time) of $T$ is the sum of the distances (or travel times) of the edges in $\mathcal{R}_T$. If the $j$-th cost is static, the cost of an edge $e$ can be obtained from function $SC^{(j)}(e)$; if the $j$-th cost is dynamic, the cost of an edge can be obtained from the GPS records in $T$ that occurred in the edge, as we did for an edge record in Section 5.3.2.

To better understand the context of trajectory $T \in \mathcal{T}_{d_m}$ and its associated preference, it is of interest to know the “shortest” routes in terms of all travel costs from the same source and destination vertices as $T$. Based on all these shortest routes, we compute an efficiency ratio vector for trajectory $T$, denoted as $ER(T) = (er^{(1)}, er^{(2)}, \ldots, er^{(N)})$, where the efficiency ratio for the $j$-th cost type is defined in Equation 5.5.

$$er^{(j)} = \frac{SR^{(j)} . c^{(j)}}{\mathcal{T}.c^{(j)}}$$ (5.5)

Here, $SR^{(j)}$ is the shortest route in terms of the $j$-th cost type and $SR^{(j)} . c^{(j)}$ is the $j$-th cost of route $SR^{(j)}$. The greater the efficiency ratio is for the $j$-th cost type, the more important the driver thinks the $j$-th cost type is.

For instance, if trajectory $T$ takes 10.2 km and 15 minutes, and the shortest and fastest routes for the same source and destination as $T$ take 8.5 km and 14.5 minutes, respectively, the efficiency ratios for the travel distance and travel time are $\frac{10.2}{8.5} = 1.2$ and $\frac{14.5}{15} = 0.97$, respectively. These ratios suggest that, in the context where $T$ occurred, the driver gave travel time higher importance than the travel distance.

Given the source and destination of trajectory $T$, the “shortest” route in terms of a static cost type is easy to compute, e.g., using Dijkstra’s algorithm on the road network where each edge is associated with the static cost. However, the “shortest” route in terms of a dynamic cost type is non-trivial to compute because dynamic cost types are time-dependent and uncertain. We discuss three different strategies to
modeling drivers’ driving behavior and thus defining the shortest route of a dynamic cost type in Section 5.4.1.2.

5.4.1.2 Driving Behavior Modeling Strategies

Assume that set $\mathbb{RS}$ contains all routes from the source of and to the destination of $T$, and let the timestamp of the first GPS record in $T$ be $t_s$. We propose three different strategies of modeling drivers’ driving behavior which are able to reduce the uncertainty and to define the “shortest” route $SR^{(j)}$ when the $j$-th cost type is dynamic.

**Expectation based Strategy (ExS):** We choose the route from $\mathbb{RS}$ with the smallest expected cost value on the $j$-th dynamic cost, denoted as $SR_{ExS}^{(j)}$, as defined in Equation 5.6.

$$SR_{ExS}^{(j)} = \text{arg min}_{R \in \mathbb{RS}} \text{EXP}(DCR^{(j)}(R, t_s)), \quad (5.6)$$

where $\text{EXP}(X)$ returns the expected value of random variable $X$. Although $ExS$ is simple and easy to be implemented, it has the disadvantage that it ignores different drivers’ distinct driving behaviors.

**Personal Ratio based Strategy (PRS):** The uncertainties of dynamic costs are largely due to different drivers exhibiting different driving behavior in different contexts. For example, aggressive drivers tend to use less time but consume more fuel compared to average drivers, and a driver who is running late may tend to drive as fast as possible.

To model a driver’s driving behavior and driving context in terms of a dynamic cost type, and thus reducing the uncertainty of the dynamic cost type for the driver in the context, we propose a concept called personal ratio. We derive a personal ratio for each trajectory made by a driver. We derive the route cost $DCR^{(j)}(R_T, t_s)$ of a trajectory $T$ made by driver $d_m$ using the instantiated dynamic cost functions and Algorithm 14.

For brevity, let $X$ denote random variable $DCR^{(j)}(R_T, t_s)$, and let $CDF(x) = P(X \leq x)$ be the cumulative distribution function for $X$. The personal ratio of trajectory $T$ with respect to the $j$-th dynamic cost and driver $d_m$ is $PR_{d_m}^{(j)}(T) = CDF(T.c^{(j)})$. The personal ratio says that driver $d_m$’s traversal of $R_T$ at time $t_s$ outperforms $(1 - PR_{d_m}^{(j)}(T)) \cdot 100\%$ percent of all traversals of route $R_T$ at time $t_s$.

Figure 5.5(a) plots the cdf of random variable $DCR^{(j)}(R_T, t_s)$, where $R_T$ is the route used by trajectory $T$. Based on its GPS records, we know that $T$ took 705 seconds, and we obtain a personal ratio of 0.72.

We can then define the shortest route $SR_{PRS}^{(j)}$ in terms of the $j$-th dynamic cost type by Equation 5.7.

$$SR_{PRS}^{(j)} = \text{arg min}_{R \in \mathbb{RS}} CDF^{-1}(PR_{d_m}^{(j)}(T)) \quad (5.7)$$
Here, \( CDF_R(\cdot) \) is the cdf of random variable \( DCR^{(j)}(R, t_s) \) which represents the cost of using route \( R \) at \( t_s \); and \( CDF_{R^{-1}}(\cdot) \) is its inverse function.

As an example, we plot the cdfs of the travel time random variables for three other routes with the same source and destination. Using the personal ratio 0.72 obtained from \( T \), we can derive travel time values for each of these routes, namely 698, 772, and 820 seconds, as shown in Fig. 5.5(b). According to Equation 5.7, the shortest travel time using \( PRS \) is then 698 seconds.

The intuition of Equation 5.7 is as follows. Consider the context in which trajectory \( T \) occurred. If the driver used a different route \( R \in \mathcal{R} \) in this context, the driver should use the same (or similar) driving behavior, and thus have the same (or similar) personal ratio, i.e., outperforming \((1 - PR^{(j)}_{dm}(T)) \cdot 100\%\) percent of the traversals of route \( R \) for the \( j \)-th dynamic cost type.

**Collective Personal Ratio based Strategy (CPRS):** This strategy considers the personal ratios of all trajectories made by driver \( dm \). Equation 5.8 defines the collective personal ratio \( CPRS \) of driver \( dm \) for the \( j \)-th dynamic cost type.

\[
\hat{PR}^{(j)}_{dm} = \arg \min_{\alpha \in [0, 1]} \sum_{T \in TR_{dm}} |\alpha - PR^{(j)}_{dm}(T)|
\]  

(5.8)

By solving Equation 5.8, we get the average personal ratio over all trajectories of driver \( dm \) as the driver’s \( CPRS \). This ratio reflects the driver’s representative driving behavior in terms of the \( j \)-th dynamic cost type.

The \( CPRS \) assumes that if driver \( dm \) used a different route \( R \in \mathcal{R} \), \( dm \) should use \( dm \)’s representative driving behavior as captured by the \( CPRS \). Thus, the \( CPRS \) returns the shortest route \( SR_{CPRS} \) according to Equation 5.9

\[
SR_{CPRS}^{(j)} = \arg \min_{R \in \mathcal{R}} CDF_{R^{-1}}(\hat{PR}^{(j)}_{dm})
\]  

(5.9)

We give the algorithmic details for computing “shortest” routes w.r.t. each dynamic cost type in Algorithm 17 (lines 21–22) in Section 5.4.2.1.
routes belong to the set of personalized skyline routes, so when we compute the personalized skyline routes, we also obtain all the “shortest” routes.

5.4.1.3 Clustering Efficiency Ratios

Recall that $\mathcal{T}_{Rd_m}$ contains all driver $d_m$’s past trajectories. For each trajectory $T \in \mathcal{T}_{Rd_m}$, an efficiency ratio vector $ER(T)$ can be derived, as described in Section 5.4.1.1.

The efficiency ratio vector of a trajectory reflects the context when the driver used the trajectory. Specifically, if the efficiency ratio of the $j$-th cost type is high, it means that the $j$-th cost of the route used by the trajectory is close to the $j$-th cost of the shortest route $SR^{(j)}$, and thus it indicates that the driver cared about the $j$-th cost in the context. In contrast, if the efficiency ratio of the $j$-th cost type is low, this indicates that the driver did not care about the $j$-th cost in the context. Thus, the trajectories that occurred in the same context should have similar efficiency ratio vectors. The idea is then that clusters of similar efficiency ratio vectors represent contexts.

Ordinal Data Transformation: Applying clustering on the efficiency ratio vectors directly has the problem that we are unable to distinguish the differences across cost types. For instance, assume that we consider travel time ($TT$) and distance ($DI$) and that we have three vectors: $ER(T_1) = (0.8, 0.6)$, $ER(T_2) = (0.8, 0.2)$, $ER(T_3) = (0.5, 0.6)$. Assume that we use an $L_p$ distance (e.g., Manhattan or Euclidean distance) when measuring the distances between vectors. Since the distance between $ER(T_1)$ and $ER(T_3)$ is less than the distance between $ER(T_1)$ and $ER(T_2)$, $ER(T_1)$ and $ER(T_3)$ are clustered together to indicate a context.

However, $ER(T_1)$ and $ER(T_2)$ should intuitively be clustered together because both indicate that the driver cares more on reducing travel time than travel distance, whereas $ER(T_3)$ indicates a context where the driver cares about the travel distance.

To capture this intuition, we transform an efficiency ratio vector into an ordinal efficiency ratio vector that orders the cost types based on how much the drivers cares about them. We then cluster these ordinal vectors to obtain contexts. The ordinal efficiency ratio vector of trajectory $T$ is denoted as $\overline{ER}(T)$. In the example, both $ER(T_1)$ and $ER(T_2)$ are $\langle TT, DI \rangle$, and $ER(T_3)$ is $\langle DI, TT \rangle$. Thus, $\overline{ER}(T_1)$ and $\overline{ER}(T_2)$ are clustered together as desired.

Clustering Ordinal Data: A naive way to clustering ordinal data is to treat each specific order as a cluster. With $N$ different cost types, we then get $N!$ different possible clusters, one per permutation of the $N$ cost types, and each such cluster reflects a context. However, a large number of cost types yields too many contexts, which is also unrealistic because drivers typically do not have large numbers of contexts: they do not consider all possible permutations over $N$ cost types.

Instead, we propose a technique that is able to identify at most $k$ important contexts (i.e., at most $k$ clusters) for a driver based on the ordinal vectors. To achieve this, we use a set called $allOrders$ to maintain all orders that appear in the ordinal
vectors of \(d_m\)'s trajectories. Each order \(ord \in \text{allOrders}\) is associated with a set of trajectories:

\[
T_{ord} = \{ T | T \in \mathbb{T}_{dm}^{d_m} \land \tilde{E}(T) = ord \}.
\]  \tag{5.10}

Continuing our example, we have \(\text{allOrders} = \{ \langle TT, DI \rangle, \langle DI, TT \rangle \} \) and \( T_{\langle TT, DI \rangle} = \{ T_1, T_2 \}, T_{\langle DI, TT \rangle} = \{ T_3 \} \).

Based on the above, we propose the clustering algorithm in Algorithm 16 which follows the agglomerative hierarchical clustering philosophy. We first initialize allOrders (lines 1–4) and then initially treat each order in allOrders as an individual cluster (line 5). Next, if the number of orders in allOrders exceeds \(k\), we combine the pair of clusters with the smallest distance (defined in Equation 5.11) into one cluster. If more than one pair of clusters has the same smallest distance, we choose the pair that has the largest number of associated trajectories. This process is applied iteratively until \(k\) clusters remain (lines 7–9). Note that if the number of orders in allOrders is less or equal than \(k\), lines 7–9 are skipped, and \(|\text{allOrders}|\) contexts, which may be fewer than \(k\) contexts, are generated for the driver.

**Algorithm 16: ContextIdentification**

\begin{algorithm}
\begin{algorithmic}
  \STATE **Input**: Driver: \(d_m\); int \(k\);
  \STATE **Output**: Contexts: \(C_1 \ldots C_l\);
  \STATE \textbf{1} Set \(\text{allOrders} \leftarrow \emptyset\);
  \FOR {each trajectory \(T \in \mathbb{T}_{dm}^{d_m}\)}
  \STATE \(\tilde{E}(T) \leftarrow \text{Transforms } E(T) \text{ into an ordinal efficiency ratio vector; \(\text{allOrders} \leftarrow \text{allOrders} \cup \tilde{E}(T)\);}\)
  \ENDFOR
  \STATE \textbf{5} Initialize \(|\text{allOrders}|\) clusters, where each cluster contains an order in set allOrders;
  \STATE \textbf{6} int \(l \leftarrow |\text{allOrders}|\);
  \WHILE {\(l > k\)}
  \STATE \textbf{7} \textbf{Combine two clusters }\(C_{l_i}\) and \(C_{l_j}\) if the distance \(\text{ClDist}(C_{l_i}, C_{l_j})\) is the smallest among all possible cluster pairs; if ties happen, choose the two clusters having the largest \(|C_{l_i}| + |C_{l_j}|\); break further ties at random;
  \STATE \textbf{8} \(l \leftarrow l - 1\);
  \ENDWHILE
  \STATE \textbf{10} \textbf{return } C_1 \ldots C_l \text{ as the } l \text{ context for driver } d_m;
\end{algorithmic}
\end{algorithm}

The distance \(\text{ClDist}(C_{l_i}, C_{l_j})\) between two clusters \(C_{l_i}\) and \(C_{l_j}\) is the largest distance between two orders in the two clusters:

\[
\text{ClDist}(C_{l_i}, C_{l_j}) = \max_{ord_x \in C_{l_i}, ord_y \in C_{l_j}} \text{SpDist}(ord_x, ord_y) \tag{5.11}
\]

Here, the distance between two orders is defined as the Spearman distance [80], which is commonly used to measure the dissimilarity between ordinal data. Assume we have \(N\) costs \(C_1, C_2, \ldots, C_N\) and each order is a permutation of these costs. If
a cost $C_i$ occurs in the $j$-th position in order $ord_x$, we say that the rank of cost $C_i$ in order $ord_x$ is $j$, denoted as $\text{rank}(C_i, ord_x) = j$. The Spearman distance between the two orders $ord_x$ and $ord_y$ is the sum of the squared differences between the ranks of all costs, as shown in Equation (5.12).

\[
\text{SpDist}(ord_x, ord_y) = \sum_{i=1}^{N} (\text{rank}(C_i, ord_x) - \text{rank}(C_i, ord_y))^2
\] (5.12)

For example, if we have two orders on three different costs $TT, FC, DI$, $ord_x = \langle TT, FC, DI \rangle$ and $ord_y = \langle DI, FC, TT \rangle$ then $\text{SpDist}(ord_x, ord_y)$ equals

\[
(1 - 3)^2 + (2 - 2)^2 + (3 - 1)^2 = 8.
\]

The number of trajectories associated with cluster $Cl_i$, denoted as $|Cl_i|$, is the number of trajectories associated with each order in the cluster: $|Cl_i| = \sum_{ord \in Cl_i} |T_{ord}|$.

The flow of the task of identifying contexts for a driver is summarized in the left side of Fig. 5.6.

---

**5.4.2 Identifying Driving Preferences**

Having found $k$ contexts for driver $d_m$, the next step is to learn a driving preference vector $w$ for each such context. We view the identification of a driving preference in a context as a classification problem, which can be solved by linear optimization. We prove that it is sufficient to consider only the personalized skyline routes when identifying driving preferences, and we propose an efficient algorithm to compute
the personalized skyline routes (in Section 5.4.2.1). We utilize the personalized skyline routes to automatically generate positive and negative training examples (in Section 5.4.2.2) for the classification problem; and we learn a specific driving preference vector by minimizing a judiciously designed objective function (in Section 5.4.2.3). The flow of identifying driving preference in a context for a driver is summarized in the right side of Fig. 5.6.

5.4.2.1 Personalized Skyline Routes

When learning the preference vector for context $C_{li}$, we consider the trajectories associated with the context, denoted as $TR_{d_m}^{C_{li}} = \bigcup_{ord \in C_{li}} T_{ord}$. Each such trajectory provides information on the preferences of driver $d_m$ in context $C_{li}$.

Consider a trajectory $T \in TR_{d_m}^{C_{li}}$. Assume that set $RS$ contains all routes with the same source and destination as $T$, and let the timestamp of the first GPS record in $T$ be $t_s$. Each route in $RS$ is associated with different static and dynamic costs. If we apply the strategies proposed in Section 5.4.1.2, dynamic costs can be reduced to a deterministic value based on the specific driver $d_m$’s driving behavior and the trip starting time $t_s$. Thus, each route $R \in RS$ is associated with a cost vector $c(R, t_s) = \langle c_1, c_2, \ldots, c_N \rangle^T$ that contains the different costs (e.g., $TT$, $DI$, and $FC$) of using route $R$ at $t_s$ by driver $d_m$.

We assume that the driver chose route $R_T$ because the route minimizes cost function $w^T \cdot c(R_T, t_s)$ among all routes in $RS$, where $w$ is the preference vector of driver $d_m$ in context $C_{li}$. In other words, every route $R \in RS$ places a constraint (shown in Equation 5.13) on the preference vector $w$.

$$w^T \cdot c(R, t_s) - w^T \cdot c(R_T, t_s) \leq 0 \quad (5.13)$$

Considering all the constraints is non-trivial because the cardinality of the route set $RS$ may be large (e.g., it may contain routes making arbitrary detours), and thus the number of constraints needed to be considered also becomes quite large. However, we do not need to consider all routes in $RS$, but only the personalized skyline routes.

Assume that we consider $N$ costs in total. If the $i$-th cost of route $R$ is smaller than the corresponding cost of $R'$ (i.e., $c(R, t_s)[i] < c(R', t_s)[i]$), we say $R$ dominates $R'$ on the $i$-th cost. If route $R$ is not dominated by $R'$ on all costs, and if $R$ dominates $R'$ on at least one cost, we say $R$ dominates $R'$, which is defined in Equation 5.14.

$$\forall i \in [1, N] \ (c(R, t_s)[i] \leq c(R', t_s)[i]) \quad \text{and}$$

$$\exists j \in [1, N] \ (c(R, t_s)[j] < c(R', t_s)[j])$$

(5.14)

Given a vector $c$, the $i$-th element of the vector is denoted as $c[i]$. 
A route $R \in RS$ is a **personalized skyline route** if there does not exist another route $R' \in RS$ such that $R'$ dominates $R$. We use $RS_{Sky}$ to denote the set of personalized skyline routes in $RS$. The personalized skyline routes satisfy an important property, as shown in Lemma 2.

**Lemma 2** Given a preference vector $w$, if route $R$ minimizes the cost function $w^T \cdot c(R, t_s)$ then $R$ must be a personalized skyline route.

**Proof 2** We prove this by contradiction. Assume that a non-skyline route $R'$ minimizes the cost function $w^T \cdot c(R', t_s)$. Since $R'$ is a non-skyline route, $R'$ must be dominated by at least one personalized skyline route $R \in RS_{Sky}$. According to Equation 5.14, for each $i \in [1, N]$, we have $w[i] \cdot c(R, t_s)[i] \leq w[i] \cdot c(R', t_s)[i]$; and there exists a $j \in [1, N]$ such that $w[j] \cdot c(R, t_s)[j] < w[j] \cdot c(R', t_s)[j]$. Thus, $w^T \cdot c(R, t_s) < w^T \cdot c(R', t_s)$, which contradicts the assumption that $R'$ minimizes the cost function because $R$ has a smaller overall cost than does $R'$.

Based on Lemma 2, we introduce Lemma 3.

**Lemma 3** When determining preference vector $w$, it is sufficient to only consider personalized skyline routes.

**Proof 3** To determine preference vector $w$, we need to make sure every route $R \in RS$ satisfies the inequality in Equation 5.13. Assuming that $R' \in RS$ is dominated by a personalized skyline route $R \in RS_{Sky}$, we have $w^T \cdot c(R, t_s) \leq w^T \cdot c(R', t_s)$, as shown in Lemma 2. Thus, if the personalized skyline route $R$ satisfies $w^T \cdot c(R, t_s) - w^T \cdot c(R', t_s) \leq 0$, this implies that $R$ satisfies $w^T \cdot c(R, t_s) - w^T \cdot c(R', t_s) \leq 0$ for each $R'$ that is dominated by $R$. Thus, we only need to ensure that every personalized skyline route satisfies Equation 5.13.

**Computing Personalized Skyline Routes:** As suggested by Lemma 3, given a trajectory $T \in TR_{C_{dm}}$, to identify the preference vector reflected by trajectory $T$, we only need to consider personalized skyline routes connecting the same source ($v_s$) and destination ($v_d$) as $T$ at time $t_s$.

Existing algorithms for computing skyline routes do not apply in our setting because they do not support dynamic costs [50] and do not consider the different driving behaviors of drivers [50, 88]. Further, due to the time-dependent and uncertain properties of dynamic costs, the graph may not necessarily be an FIFO graph. In contrast, most existing work on time-dependent and uncertain routing assumes FIFO graphs [84]. We propose an efficient algorithm to compute personalized skyline routes $RS_{Sky}$ that is able to support dynamic costs and to consider different drivers’ driving behaviors using the driving behavior modeling strategies proposed in Section 5.4.1.2.

---

5We use “personalized skyline route” instead of “skyline route” because when deriving the cost vector for a route, we consider driver $d_m$’s driving behavior which is quantified by $d_m$’s personal ratio.
5.4. IDENTIFYING CONTEXT-AWARE DRIVING PREFERENCES

A naive way to compute personalized skyline routes is to enumerate all routes from $v_s$ to $v_d$ and then to compute, for each route, the random variables for the dynamic costs, applying one of the strategies proposed in Section 5.4.1.2 to derive a deterministic value for the dynamic cost. Finally, dominance checking is applied to the cost vectors of the routes. This approach is very inefficient and works only for very small road networks.

To facilitate efficient personalized skyline route computation, for each edge $e_k$, we maintain a minimum cost vector $e_k.MCV = \langle mc^{(1)}, \ldots, mc^{(N)} \rangle$. In particular, consider edge $e_k = (v_i, v_j)$. For a static cost, the minimum cost value on $e_k$ is the static cost of $(v_j, v_i)$. For a dynamic cost, the minimum cost value maintained on $(v_i, v_j)$ is the minimum value on edge $(v_j, v_i)$. For instance, $e_k.mc^{(D)} = 1.3$ if the length of $(v_j, v_i)$ is 1.3 km; and $e_k.mc^{(T)} = 30$ if it takes at least 30 seconds to traverse $(v_j, v_i)$ no matter who drives the vehicle and no matter whether the traversal occurs during peak or off-peak hours. This information can be obtained from the GPS data on edge $(v_j, v_i)$.

Based on the above, Algorithm 17 starts by issuing $N$ one-to-all shortest path queries (using Dijkstra’s algorithm in function \texttt{OneToAllSP}) from the destination based on the edges’ minimum cost vectors (lines 2–5). Each query considers a travel cost type of interest. After this, each vertex is associated with a minimum cost vector, based on the edges’ minimum cost vectors (lines 2–5). Each query considers a travel cost type (e.g., the $i$-th), the possible “shortest” route $R^{(i)}_{\text{next}}$ from the source to the destination is also identified by \texttt{OneToAllSP}. We call function \texttt{ComputeCostVector} (Algorithm 18) to compute the cost vector for each route $R^{(i)}_{\text{next}}$. For a static cost, the cost of $R^{(i)}_{\text{next}}$ is the sum of the costs of the edges in $R^{(i)}_{\text{next}}$. For a dynamic cost, we compute the corresponding random variable of $R^{(i)}_{\text{next}}$, based on Algorithm 14, and we apply one of the three strategies from Section 5.4.1.2 to derive a deterministic value. After that, we obtain cost vector $c(R^{(i)}_{\text{min}}, t_s)$ for route $R^{(i)}_{\text{min}}$.

We use a set $\mathbb{R}^{SKY}$ to maintain the personalized skyline routes that we identified so far. After computing the cost vector of route $R^{(i)}_{\text{min}}$, we update set $\mathbb{R}^{SKY}$. Specifically, if a route in $\mathbb{R}^{SKY}$ dominates $R^{(i)}_{\text{min}}$, $\mathbb{R}^{SKY}$ needs not be updated. Otherwise, $R^{(i)}_{\text{min}}$ is added to $\mathbb{R}^{SKY}$, and any routes already in $\mathbb{R}^{SKY}$ that are dominated by $R^{(i)}_{\text{min}}$ are deleted. This procedure is conducted by function \texttt{UpdateSkyline}.

Next, we explore the possible routes from the source to the destination (lines 6–20). For a partially explored route $R_{\text{next}}$, we compute the cost vector $c(R_{\text{next}}, t_s)$ using function \texttt{ComputeCostVector}. If $R_{\text{next}}$ stops at vertex $v'$, the minimum cost vector $v'.MCV$ records the minimum costs from $v'$ to the destination $v_d$. Next, vector $EstCostVector$ sums vectors $c(R_{\text{next}}, t_s)$ and $v'.MCV$, and this vector records the minimum costs of routes connecting $v_s$ and $v_d$ while using $R_{\text{next}}$ from $v_s$ to $v'$. If vector $EstCostVector$ is dominated by the cost vector of a route $R'$ in $\mathbb{R}^{SKY}$,
Algorithm 17: ComputingSkylineRoutes

**Input** : Source: $v_s$; Destination: $v_d$; Time: $t_s$; Strategy: $x$; 
**Output**: SkylineRoutes: $RS_{Sky}$;

1. SkylineRoutes $RS_{Sky} \leftarrow \emptyset$;
2. for each $1 \leq i \leq N$ do
   3. $R_{min}^{(i)} \leftarrow \text{OneToAllSP}(v_d, i)$;
   4. $c(R_{min}^{(i)}, t_s) \leftarrow \text{ComputeCostVector}(R_{min}^{(i)}, t_s, x)$;
   5. $RS_{Sky} \leftarrow \text{UpdateSkyline}(RS_{Sky}, R_{min}^{(i)})$;
6. Initialize a queue $Q$, and $Q$.enqueue($(v_s, v_s)$);
7. repeat
   8. $R_{next} \leftarrow Q$.dequeue();
   9. $v \leftarrow$ the last vertex of $R_{next}$;
   10. for each $e_k \in E$ with $e_k.s = v$ and $e_k$ is not in $R_{next}$ do
       11. $R_{next} \leftarrow \text{Append } e_k \text{ to } R_{next}$;
       12. $c(R_{next}, t_s) \leftarrow \text{ComputeCostVector}(R_{next}, t_s, x)$;
       13. $v' \leftarrow e_k.d$;
       14. EstCostVector $\leftarrow c(R_{next}, t_s) + v'.MCV$;
       15. if $v' \neq v_d$ then
           16. if EstCostVector is not dominated by any route in $RS_{Sky}$ then
               17. Q.enqueue($R_{next}$);
           else
               18. $RS_{Sky} \leftarrow \text{UpdateSkyline}(RS_{Sky}, R_{next})$;
       end if
   end for
   19. until $Q$ is empty;
20. for each $1 \leq i \leq N$ do
   21. $SR^{(i)}_c(i) = \min_{R \in RS_{Sky}} (c(R, t_s)[i])$;
22. return $RS_{Sky}$;

there is no need to keep exploring $R_{next}$ because all further routes will be dominated by $R'$. If not, $R_{next}$ needs to be explored further, and is added to query $Q$. The procedure stops when $Q$ is empty, and the routes in $RS_{Sky}$ are the personalized skyline routes.

In an FIFO graph, assume that routes $R_1$ and $R_2$ exist from the source to an intermediate vertex $v$ enroute to the destination. Route $R_1$ can be omitted if it is stochastically dominated by $R_2$ because a route from the source to the destination that uses $R_2$ as its pre-route will always stochastically dominate one that uses $R_1$ as its pre-route \[84\].

In contrast, in a non-FIFO graph, $R_1$ cannot be omitted because a route that uses $R_1$ as its pre-route may stochastically dominate a route using $R_2$ as its pre-route. Thus, to ensure that we do not miss any skyline routes when given a non-FIFO graph, Algorithm 17 keeps all partially explored routes in queue $Q$ if they (according to
their estimated “best” possible costs to the destination) are not dominated by existing candidate skyline routes.

Finally, we compute the minimum cost values for all cost types, which are used for computing efficiency ratios (lines 21-22).

![Algorithm 18: ComputeCostVector](image1)

![Algorithm 19: UpdateSkyline](image2)

5.4.2.2 Choosing Positive vs. Negative Skyline Routes

According to the preference model, a driver chose a route $R_T$ because it minimizes the cost function among all possible routes in $R$. Thus, route $R_T$ should be a personalized skyline route. However, when studying trajectory data, we find that drivers also follow routes that are not personalized skyline routes. This can happen because (i) the driver did not know that a “better” route (e.g., a personalized skyline
route) existed or (ii) the driver did not mind to choose a sub-optimal route, e.g., a route that has the second or third smallest value based on the cost function. In other words, ideally, a driver should choose personalized skyline routes, but sometimes drivers also choose non-skyline routes.

When a driver chooses a non-skyline route $R_T$, we try to map $R_T$ to a set of positive skyline routes that reflect the driver’s preferences. The intuition of using positive skyline routes is that the driver should use one of them if the driver knows costs of all routes (cf. reason (i)) and the driver chooses the route that minimizes the cost function (cf. reason (ii)).

According to the proof of Lemma 3, given any preference vector, the value of the cost function on a personalized skyline route is always smaller than that of a route that is dominated by the personalized skyline route. If a personalized skyline route dominates the non-skyline route $R_T$, the skyline route should have a smaller value for the cost function no matter which preference vector the driver has. In this sense, the personalized skyline route better reflects the driver’s preferences. Based on the above, the positive skyline routes of $R_T$ are the personalized skyline routes that dominate $R_T$: $RS_{Sky}^{pos}(R_T) = \{R | R \in RS_{Sky}, R \text{ dominates } R_T\}$.

If a skyline route does not dominate the non-skyline route $R_T$, it is possible that the value of the cost function on route $R_T$ is smaller than that on the skyline route for some preference vectors. Thus, such skyline routes cannot reflect the driver’s preference. Based on this, the remaining skyline routes are defined as the negative skyline routes of $R_T$: $RS_{Sky}^{neg}(R_T) = RS_{Sky} \setminus RS_{Sky}^{pos}(R_T)$.

Figure 5.7 shows an example when considering $TT$ and $FC$. Routes $R_1$, $R_2$, $R_3$, and $R_4$ are skyline routes, and $R_T$ is the route actually used by the driver. Since $R_2$ and $R_3$ dominate $R_T$, both routes are positive skyline routes of $R_T$; and $R_1$ and $R_4$ are negative skyline routes of $R_T$.

![Figure 5.7: Positive and Negative Skyline Routes of Route $R_T$](attachment:figure57.png)
5.4. IDENTIFYING CONTEXT-AWARE DRIVING PREFERENCES

5.4.2.3 Identifying Preference Vectors

Given a positive skyline route \( R_i \in \mathbb{R}^{Sky}_{pos}(R_T) \) and a negative skyline route \( R_j \in \mathbb{R}^{Sky}_{neg}(R_T) \), we derive two feature vectors \( f_{i,j} = c(R_i, t_s) - c(R_j, t_s) \) and \( f_{j,i} = c(R_j, t_s) - c(R_i, t_s) \). The two feature vectors satisfy the following constraints \( w^T \cdot f_{i,j} < 0 \) and \( w^T \cdot f_{j,i} > 0 \).

These two constraints motivate us to transform the problem of identifying the preferences \( w \) to solving a classification problem with binary class labels, e.g., good and bad. Specifically, we regard \( f_{i,j} \) as a training feature for class good and \( f_{j,i} \) as a training feature for class bad. For each pair of a positive and a negative skyline route, we are able to generate two training features, one for class good and one for class bad.

Based on the generated features and their associated class labels, the target is to learn a linear classification \( w \) such that \( w^T \cdot f < 0 \) if \( f \) is a training feature for class good; and \( w^T \cdot f > 0 \) if \( f \) is a training feature for class bad. The linear classification is a hyperplane with norm \( w \) that separates the good training features from the bad training features.

![Identifying Preference Vector w Using Classification](image.png)

Figure 5.8: Identifying Preference Vector \( w \) Using Classification

Consider the case shown in Fig. 5.7. As there are two positive and two negative skyline routes, we obtain four good training features and four bad training features after we pair each positive skyline route and each negative skyline route. See Fig. 5.8(a).

Figure 5.8(a) concerns a 2D space involving travel time and fuel consumption, so a 2D hyperplane is a line. Note that more than one classifier (i.e., line) is able to separate the good and the bad training features. For example, both lines shown in Fig. 5.8(a) separate the good and bad features.

Motivated by the basic idea of Support Vector Machines (SVMs), we are interested in getting an optimal hyperplane that satisfies the property that the closest
distance from a good training feature to it equals the closest distance from a bad training feature to it, such that this distance is maximized.

Expressing this idea mathematically, we have \( w^T \cdot f \leq -c \) (where \( c > 0 \)) for good training features and \( w^T \cdot f \geq c \) for bad training features, and the closest distance from a good (or a bad) training feature to the optimal hyperplane is \( \frac{c}{||w||} \). Fig. 5.8(b) exemplifies this. In order to obtain the optimal hyperplane, we need to maximize the distance \( \frac{c}{||w||} \), which is equivalent to minimizing \( ||w|| \) because \( c \) is a positive value.

In addition, we constrain all the values in preference vector \( w \) to be non-negative (see Inequality 5.16). A negative preference on a cost renders a route with a higher value on the cost more desirable, which is counter-intuitive. Since \( w \) is a non-negative vector, minimizing \( ||w|| \) is equivalent to minimizing \( \sum_{i=1}^{N} w[i] \) subject to several constraints, as detailed below.

\[
\text{minimize} \quad \sum_{i=1}^{N} w[i] \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \ \
$CL_i$ has trajectories associated with it, denoted as $\bigcup_{ord \in CL_i} T_{ord}$, where $T_{ord}$ is defined in Equation 5.10. We define the temporal aspect of a trajectory as an interval that extends from the start time to the end time of the trajectory. The temporal aspect of cluster $CL_i$ is the union of the temporal aspects of all the trajectories associated with $CL_i$.

When a driver plans a trip at time $t_q$, we choose the context $CL_i$ whose temporal aspect covers $t_q$. If more than one context’s temporal aspect covers $t_q$, we randomly choose one. If no contexts’ temporal aspect covers $t$, we choose the context whose temporal aspect is closest to $t$. After identifying an appropriate context $CL_i$ for the driver, the corresponding preference $w$ is also identified. Finally, we call Algorithm 17 to compute the skyline routes, and we return the personalized skyline route $R$ that minimizes $w^T \cdot c(R, t_q)$.

5.5 Empirical Studies

We report on empirical studies of the effectiveness and efficiency of the proposed techniques.

5.5.1 Experimental Setup

**GPS Trajectories:** We use more than 180 million GPS records collected at 1 Hz (i.e., one GPS record per second) in Denmark from 183 drivers during week days in 2007 and 2008. The GPS records are from an experiment where young drivers start out with a rebate on their car insurance and then are warned if they speed and are penalized financially if they continue to speed.

The GPS records are map matched [67] to OpenStreetMap’s road network for Denmark, which consists of some 668K vertices and 1,623K edges. Next, data cleaning is conducted to exclude outlier GPS records that have unreasonable speeds (i.e., higher than 300 km/h). After map-matching and data cleaning, we obtain approximately 623K trajectories. We split the trajectories into a training set with trajectories collected from April 2007 to September 2007 and a testing set with trajectories collected from October 2007 to March 2008.

**Travel Costs:** We consider three travel costs: travel distance ($DI$), travel time ($TT$), and fuel consumption ($FC$), where $DI$ is static and $TT$ and $FC$ are dynamic. We do not consider other costs such as the number of traffic lights because OpenStreetMap for Denmark does not provide such information. However, the proposed techniques can support an arbitrary number of costs.

The travel distances of edges are computed based on the coordinates of the corresponding vertices that are recorded in OpenStreetMap. Travel times are obtained as the difference between the times of the last and first GPS records of trajectories on an edge. We use the SIDRA-running model [39] to compute fuel consumption based on the available GPS records. A recent benchmark [39] indicates that SIDRA-running is appropriate for this purpose.
Some edges are not covered by any GPS records. For such edges, a travel-time value is derived by dividing the length of the edge by the speed limit of the edge; and a fuel consumption value is estimated using the SIDRA-Running model based on the length and speed limit of the edge. Speed limits are obtained from OpenStreetMap.

Next, as an edge’s $TT$ and $FC$ weights, we generate Gaussian random variables with the derived cost value as the mean and one fifth of the mean as the standard deviation.

**Parameters:** We vary some parameters according to Table 5.3 where default values are shown in bold. Parameter $MLT$ is the mean lifetime used in Equation 5.4. Among the three driving behavior modeling strategies proposed in Section 5.4.1.2, CPRS is the default strategy because it produces the best accuracy (detailed in Section 5.5.3). When learning the driving preference, we vary the number of training trajectories per driver from 20 to 200, with 100 as the default number. We also vary the number of contexts from 1 to 3.

**Table 5.3: Parameter Settings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MLT$</td>
<td>5, 15, 30 (minutes)</td>
</tr>
<tr>
<td>Driving behavior modeling strategies</td>
<td>$ExS, PRS, CPRS$</td>
</tr>
<tr>
<td>Training trajectories per driver</td>
<td>20, 50, 100, 150, 200</td>
</tr>
<tr>
<td>Number of contexts</td>
<td>1, 2, 3</td>
</tr>
</tbody>
</table>

**Implementation Details:** The proposed algorithms are implemented in Java using JDK 1.7. To ease the management of Gaussian mixture models, the jEMF package is applied. The linear optimization algorithm is implemented with the help of the Apache Commons Mathematics Library and standard slack variables are implemented to deal with non-linearly separable cases. When implementing Algorithm 17, we only consider the vertices whose distances to the destination are smaller than 2 times the shortest distance between source and destination. A computer with Debian 3.2.46-1, a 16–core Intel Xeon E5-2650 @ 2.00 GHz CPU, and 32 GB main memory is used for all experiments.

### 5.5.2 Accuracy of Dynamic Costs

We consider the discrete approach ($DA$) and the continuous approach ($CA$) described in Section 5.3.2. We use 96 15-minute intervals per day, i.e., setting $\alpha = 15$ minutes, which is typically used as the finest time granularity in the transportation area. We consider the three mean lifetime ($MLT$) parameter settings shown in Table 5.3.

---

http://www.lix.polytechnique.fr/~nielsen/MEF/
http://commons.apache.org/proper/commons-math/
5.5. EMPIRICAL STUDIES

We use the training trajectory set to learn the TT and FC dynamic costs using DA and CA, respectively. We use the testing trajectory set to evaluate the accuracy of the learned dynamic costs.

In the first set of experiments, the accuracy of an obtained dynamic cost of an edge $e_k$ is evaluated by the likelihood ratio \[61\], defined as follows.

\[
\text{LikelihoodRatio} = \prod_{l_i \in \text{testing}} \frac{CA(l_i.e, l_i.t, l_i.cost)}{DA(l_i.e, l_i.t, l_i.cost)},
\]

where $l_i$ is a cost record in a testing trajectory, $CA(l_i.e, l_i.t, l_i.cost)$ returns the likelihood of observing cost $l_i.cost$ (i.e., a travel time or a fuel consumption value) on edge $l_i.e$ at time $l_i.t$ based on the dynamic weights obtained from CA, and $DA(l_i.e, l_i.t, l_i.cost)$ returns the corresponding likelihood using the dynamic weights obtained from DA. Since we use Gaussian mixture models to represent random variables, the likelihoods are the densities of the Gaussian mixture models.

The likelihood ratios are reported in Fig. 5.9. When the ratio exceeds 1, CA has better accuracy; and the larger the ratio, the better the accuracy of CA compared to that of DA.

![Figure 5.9: Likelihood Ratios of Instantiated Dynamic Costs](image)

Figure 5.9 suggests that CA outperforms DA on most edges for both TT and FC, and it suggests that a mean lifetime of 15 yields the best overall accuracy. Specifically, for TT, CA with $MLT = 5$, $MLT = 15$, and $MLT = 30$ outperform DA on 64.3%, 82.5%, and 79.3% of edges, respectively; and for FC, CA with $MLT = 5$, $MLT = 15$, and $MLT = 30$ outperform DA on 80.3%, 86.1%, and 80.7% of edges, respectively. When $MLT = 5$, the decaying function used in CA only considers very close-by intervals, and thus its accuracy is similar to that of DA and has the lowest accuracy. When considering the ranges where the likelihood ratios exceed 1, i.e., $e^0$, the setting with $MLT = 15$ has higher percentages of edges than that of $MLT = 30$, although the setting with $MLT = 30$ has a slightly higher bar in the last range. This suggests that CA with $MLT = 15$ has the best accuracy.
In the second set of experiments, the accuracy of an obtained dynamic cost of an edge $e_k$ is evaluated by the average Kullback-Leibler divergence (KL-divergence), defined as follows.

$$\text{AvgKL} = \frac{1}{|NEI|} \sum_{i \in NEI} KL\left(\text{TrainD}(e_k, i), \text{TestD}(e_k, i)\right)$$

Here, $NEI$ indicates the set of non-empty intervals of edge $e_k$. An interval is non-empty if more than one trajectory exists in the testing trajectory set that occurred in the interval on edge $e_k$. $KL(\cdot, \cdot)$ returns the KL-divergence between two distributions. Next, $\text{TestD}(e_k, i)$ is the actual travel cost distribution on edge $e_k$ during interval $i$, which is derived from the testing trajectory set. Finally, $\text{TrainD}(e_k, i)$ is the estimated travel cost distribution on edge $e_k$ during interval $i$, which is derived from the training trajectory set. We consider four different approaches to deriving $\text{TrainD}$, namely $\text{DA}$, $\text{CA}$ with $\text{MLT} = 5$, $\text{CA}$ with $\text{MLT} = 15$, and $\text{CA}$ with $\text{MLT} = 30$.

The average KL-divergence is reported in Fig. 5.10. The smaller the average KL-divergence is, the more similar the actual distribution and the estimated distribution are. The figure suggests that all $\text{CA}$, regardless of the $\text{MLT}$ parameter setting, outperform $\text{DA}$—$\text{DA}$ has the lowest bars when the average KL-divergence is low, but has the highest bar when the average KL-divergence is large. Further, among the three $\text{CA}$ variants, the figure suggests that $\text{CA}$ with $\text{MLT} = 5$ has the worst accuracy and that $\text{CA}$ with $\text{MLT} = 15$ has the best accuracy, which is consistent with the results using likelihood ratios shown in Fig. 5.9.

Both Fig. 5.9 and Fig. 5.10 suggest that $\text{CA}$ with time decaying captures the dynamic travel costs more accurately, and the indicate that $\text{CA}$ with $\text{MLT} = 15$ has the best accuracy. Thus, we use $\text{CA}$ with $\text{MLT} = 15$ in subsequent experiments.
5.5. EMPIRICAL STUDIES

5.5.3 Accuracy of Preferences

To observe the accuracy of the obtained driving preference vector $w$, we choose 100 testing trajectories for each driver. To make the accuracy study as meaningful as possible, we make sure that (1) the testing trajectories have different source-destination pairs; (2) the source-destination pairs of the testing trajectories are different from those of the drivers’ training trajectories.

The intuition of the accuracy study is that if vector $w$ accurately reflects the drivers’ driving preferences, the actual routes used by a driver should be similar to the routes obtained using $w$.

Given a testing trajectory, we denote the actual route of the testing trajectory by $AR$, and we denote the recommend route by $RR$. We use the Jaccard coefficient (Equation 5.20) between the set of edges in $AR$ and the set of edges in $RR$ to measure the similarity between the two routes.

$$\text{Jac}(AR, RR) = \frac{|\{AR\text{edges}\} \cap \{RR\text{edges}\}|}{|\{AR\text{edges}\} \cup \{RR\text{edges}\}|} \tag{5.20}$$

For example, if $AR = \langle e_1, e_2, e_3 \rangle$ and $RR = \langle e_1, e_4 \rangle$, we have $\text{Jac}(AR, RR) = \frac{|\{e_1\}|}{|\{e_1, e_2, e_3, e_4\}|} = 0.25$.

Jaccard similarity may not accurately reflect how similar the two routes of interest are because it treats every edge equally. Continuing the previous example, if the lengths of edges $e_1$, $e_2$, $e_3$, and $e_4$ are 100, 2, 3, and 7, respectively, the two routes are actually quite similar because they share a very long edge in the beginning and only deviate slightly at the end. To better capture this aspect, we also introduce a weighted Jaccard coefficient that treats longer edges as more important, as shown in Equation 5.21.

$$\text{WJac}(AR, RR) = \frac{\sum_{e \in \{AR\text{edges}\} \cap \{RR\text{edges}\}} \text{len}(e)}{\sum_{e \in \{AR\text{edges}\} \cup \{RR\text{edges}\}} \text{len}(e)}, \tag{5.21}$$

where $\text{len}(e)$ returns the length of edge $e$. Continuing the previous example, we have $\text{WJac}(AR, RR) = \frac{100}{100 + 2 + 3 + 7} = 0.89$.

When $AR$ and $RR$ are the same, the Jaccard and weighted Jaccard coefficients are both 1; and when $AR$ and $RR$ do not share any edges, both coefficients are 0. The higher the (weighted) Jaccard coefficient is, the more similar the actual and the recommend routes are.

We consider two baselines that are used extensively in existing navigation services. The first recommends the shortest route, denoted as $BR_1$. The second recommends the fastest route using the speed limits, denoted as $BR_2$. This second baseline is regarded as the state-of-the-art approach since major navigation services tend to recommend fastest routes [11, 92].

*Here, the weights refer to the lengths of edges. Alternative weights may refer to the travel times or the fuel consumption of edges. We do not consider them in the experiments because both travel time and fuel consumption of edges are uncertain, rendering them unattractive for use as weights.
To quantify the advantages of the recommended route using the learned driving preferences over the recommend routes using the baseline methods, we define an improvement ratio ($IR$) based on Jaccard coefficients, in Equation 5.22, and a weighted improvement ratio ($WIR$) based on weighted Jaccard coefficients, in Equation 5.23.

$$IR(AR, BR_i) = \frac{Jac(AR, RR)}{Jac(AR, BR_i)} \tag{5.22}$$

$$WIR(AR, BR_i) = \frac{WJac(AR, RR)}{WJac(AR, BR_i)} \tag{5.23}$$

When the (weighted) improvement ratio exceeds 1, the recommend route $RR$ using the learned driving preference vector is more similar to the actual route than the shortest ($BR_1$) or fastest ($BR_2$) route.

The distribution of the improvement ratios and weighted improvement ratios on all testing trajectories w.r.t. the shortest routes are reported in Figs. 5.11 and 5.12, respectively. In the best setting (using CPRS and 3 contexts), our proposal outperforms $BR_1$ for 92% and 96% of the testing trajectories using improvement ratios and weighted improvement ratios, respectively; and on average, our proposal achieves improvement ratios of 4.60 and weighted improvement ratios of 7.47 w.r.t. $BR_1$, respectively.

Similar results w.r.t. the fastest routes can be observed in Figs. 5.13 and 5.14. In particular, in the best setting, our proposal outperforms $BR_2$ for 73% and 75% of the testing trajectories using improvement ratios and weighted improvement ratios, respectively; and on average, our proposal achieves improvement ratios of 2.68 and weighted improvement ratios of 6.36 w.r.t. $BR_2$. These results suggest that our proposal is effective.

![Figure 5.11: Improvement Ratios w.r.t. Shortest Routes](image)

(a) With 1 context  (b) With 2 contexts  (c) With 3 contexts

Figure 5.11: Improvement Ratios w.r.t. Shortest Routes

When fixing the number of contexts, CPRS has the lowest bars in the bar groups on the left side of the vertical line, i.e., with (weighted) improvement ratios below 1, and ExS has the highest bars; also, CPRS has the highest bars in the bar groups on the right side of the vertical line, i.e., with (weighted) improvement ratios above 1, and ExS has the lowest bars.

These findings suggest that CPRS outperforms the other two strategies and that ExS has the worst performance. This is because ExS uses the expectation for all
### 5.5. Empirical Studies

#### Figure 5.12: Weighted Improvement Ratios w.r.t. Shortest Routes

(a) With 1 context  
(b) With 2 contexts  
(c) With 3 contexts

#### Figure 5.13: Improvement Ratios w.r.t. Fastest Routes

(a) With 1 context  
(b) With 2 contexts  
(c) With 3 contexts

#### Figure 5.14: Weighted Improvement Ratios w.r.t. Fastest Routes

(a) With 1 context  
(b) With 2 contexts  
(c) With 3 contexts
drivers and does not distinguish among different drivers. Next, \textit{PRS} uses the personal ratio of each individual training trajectory to derive values for the dynamic cost types, which sometimes may be overly specific to an individual training trajectory. Finally, \textit{CPRS} uses a personal ratio that is derived from all training trajectories, and it best captures an individual driver’s behavior.

Next, when fixing the strategy, having more than one context is better than having only one context. Given a strategy, its corresponding bars on the left side of the vertical line become lower, and its corresponding bars on the right side of the vertical line become higher, as the number of context increases. This suggests that drivers tend to have different driving preferences in different contexts and that our proposals are able to identify such contexts and corresponding driving preferences.

Figure 5.15 shows two case studies, where the dark, bold routes are the actual routes used by testing trajectories; the blue routes are the recommend routes based on the learned driving preferences; the red routes are the shortest routes; and the green routes are the fastest routes.

In the first case, the shortest route is quite different from the actual route, and the (weighted) improvement ratio w.r.t. the shortest route is much higher than 1. The (weighted) improvement ratio w.r.t. the fastest route also exceeds 1 because the deviation between the blue route and the black route in the lower, right part is smaller than the deviation between the green route and the black route in the upper, left part. In the second case, the recommend route is almost the same as the actual route, whereas both the shortest route and the fastest route only partially overlap with the actual route.

Both cases also suggest that the fastest routes tend to be more similar to the actual routes than the shortest routes. This is also seen in Figs. 5.11—5.14 where the (weighted) improvement ratios w.r.t. the shortest routes ($BR_1$) are higher than those for the fastest routes ($BR_2$).

Since \textit{CPRS} performs the best, and the results using improvement ratios and
weighted improvement ratios are consistent, we use only CPRS and weighted improvement ratios in the following experiments.

In the next experiment, we group testing trajectories according to the drivers who produced the trajectories, and we study the accuracy of the obtained driving preference vectors on a per driver basis. For each driver, we compute the percentage of superior trajectories w.r.t. all the driver’s testing trajectories. A testing trajectory is a superior trajectory if the corresponding weighted improvement ratio exceeds 1. This indicates that the recommended route using the obtained driving preference \( w \) is more similar to the actual route used by the trajectory than the baseline routes.

We report the percentage of superior trajectories per driver in Fig. 5.16. Most drivers have a percentage of superior trajectories that exceeds 50% (i.e., on the right side of the vertical line), indicating that there are more cases where the recommended routes are better than the baseline routes. As the number of contexts increases, the percentage of superior trajectories also increases. In particular, when using 3 contexts, 97.2% (87.5%) of drivers have at least 50% (70%) of superior trajectories w.r.t. the shortest routes; and 91.7% (51.5%) of drivers have at least 50% (70%) superior trajectories w.r.t. the fastest routes. This suggests that the context-aware driving preferences are effective at providing better routes.

We proceed to vary the number of training trajectories per driver according to Table 5.3—see Fig. 5.17. As the number of training trajectories increases, the average weighted improvement ratios of all testing trajectories also increase. This suggests that when more training trajectories are used, the accuracy of the obtained driving preferences increases. The increasing trend is more obvious when the number of training trajectories is small, e.g., when going from 20 to 100. When having more than 100 training trajectories, the benefit of additional training trajectories is less clear.

Figure 5.17 also suggests that having more than one context is better than having only one context. However, having three contexts is only slightly better than having
two contexts, which suggest that drivers in the GPS data set typically have at least two different contexts and may not have too many contexts.

5.5.4 Efficiency

The proposal includes an off-line training phase and an on-line querying phase, where only the latter is time-critical.

Off-line training has three major parts: (i) context identification using Algorithm 16; (ii) the computation of personalized skyline routes for training trajectories using Algorithm 17; and (iii) the learning of preference vectors (i.e., solving the optimization shown in Equation 5.15). Of these, part ii is the most time-consuming. Table 5.4 reports the average run-time of identifying personalized skyline routes for a training trajectory using Algorithm 17. Since both PRS and CPRS need to compute the inverse CDF of a random variable, while ExS only computes the expectation of a random variable, PRS and CPRS take longer than ExS.

Table 5.4: Average Run-Time, Training Phase

<table>
<thead>
<tr>
<th></th>
<th>ExS</th>
<th>PRS</th>
<th>CPRS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>2,813 ms</td>
<td>3,362 ms</td>
<td>2,844 ms</td>
</tr>
</tbody>
</table>

The total run-time of the training phase depends on the size of the set of training trajectories. The more training trajectories we use, the longer the run-time. In the most time-consuming setting, we use 200 training trajectories per driver, and the total run-time per driver is 200 times the average run-time per training trajectory, depending on the chosen strategies. On average, part ii uses around 33 minutes per driver to identify the skyline routes for 200 training trajectories using all three strategies. This part can easily be parallelized among different drivers because the identifica-
tion of skyline routes for different drivers is independent. In the experiments, we use multiple (up to 8) processes to identify skyline routes for different drivers in parallel, which significantly reduces the total run-time of part ii.

Compared to part ii, the run-times of parts i and iii are negligible. For both parts, the most time-consuming setting is with 3 contexts and 200 training trajectories per driver. In this setting, the context identification part takes less than 1 second per driver on average, and the preference learning part takes less than 27 seconds per driver on average.

Based on the obtained driving preferences, we expect to recommend routes to drivers based on their driving preferences in real-time. Since CPRS produces the best results, we use CPRS in Algorithm 17 in the testing phase. When using different numbers of contexts, the average run-time of computing a recommended route is reported in Table 5.5.

<table>
<thead>
<tr>
<th></th>
<th>1 Context</th>
<th>2 Contexts</th>
<th>3 Contexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Run-Time</td>
<td>2,874 ms</td>
<td>3,020 ms</td>
<td>2,977 ms</td>
</tr>
</tbody>
</table>

The run-times for the cases having more than one context exceed the case of having only one context because it takes additional time to identify appropriate contexts and thus to choose appropriate preferences. Recommending routes based on the obtained driving preferences takes around 3 seconds on average. In the testing phase of our experiments, most of the routes can be recommend within 2 seconds, and it takes at most 5 seconds, which is acceptable for on-line use.

5.5.5 Summary

Three important observations follow from the empirical studies. First, the continuous approach (CA) captures the dynamic travel costs (i.e., travel times and fuel consumptions) more accurately than does the discrete approach (DA). Second, the proposed context identification and driving preference learning methods are able to identify distinct contexts for each driver; and they make it possible to identify a driving preference for each context for each driver from the driver’ historical trajectories. As a result, they enable effective context-aware and personalized routing. Third, the run-time of the context-aware and personalized routing is acceptable for on-line use.

5.6 Related Work

The coverage of related work is organized according to key themes related to the paper’s contributions.
ROUTE RECOMMENDATION USING TRAJECTORIES: Two recent works \[29, 55\] study how to identify frequent routes from trajectories. Another study \[87\] focuses on deriving accurate travel times (fuel consumption) from trajectories, thus providing fastest (most eco-friendly) routes. These approaches do not take into account the distinct preferences of different drivers, and they can only provide the same (frequent or fastest or most eco-friendly) routes to all drivers. In contrast, our proposal is able to provide personalized routes to different drivers based on the learned driving preferences.

DERIVING DYNAMIC COSTS: Two recent works \[88, 92\] study how to derive dynamic costs from GPS trajectories. In particular, one study employs histograms to represent the distributions of travel times \[92\]. Instead, we use Gaussian mixture models (GMMs), which are continuous and parametric, to represent the distributions of dynamic costs. This has two benefits. First, the distributions of dynamic costs are essentially continuous, whereas histograms are good at representing discrete distributions, which are not well suited for the accurate modeling of continuous distributions \[88\]. Second, the use of GMMs contributes to making route cost computations (i.e., Algorithm 14) efficient because integrals can be computed efficiently based on the relatively few parameters in GMMs. In contrast, the use of histograms makes route cost computation inefficient, especially for long routes, because very large numbers of histogram buckets are involved in the computation.

A discrete approach to deriving dynamic costs using GMMs is proposed in a recent study \[88\]. A detailed coverage of how our continuous approach improves on the discrete approach is given in Section 5.3.2. The experimental results in Section 5.5.2 indicate that the continuous approach outperforms the discrete approach.

DRIVING BEHAVIOR MODELING: Two works \[52, 92\] study the driving behaviors of drivers. However, both only support one travel cost—travel time; and they do not distinguish among different contexts. In contrast, our proposal supports an arbitrary number of travel costs and multiple contexts. Further, one of the two studies \[52\] only works for roads that a driver has used previously, while our proposal works for all roads.

STOCHASTIC SKYLINE ROUTE COMPUTATION: The computation of skyline routes based on a set of static costs has been the subject of one previous study \[50\]. However, the proposed techniques are not applicable in our setting, which considers dynamic costs. A few studies have been reported that do consider dynamic costs \[28, 33, 58, 62, 63, 78\]. However, these studies only consider one type of dynamic cost, and they thus cannot be applied in our setting where multiple dynamic travel costs are considered.

A recent study \[88\] proposes an efficient stochastic skyline route computation method while considering dynamic costs, but it does not consider the different driving behaviors of drivers, and thus it does not apply in our setting. In that study, the cost of a route is a vector of random variables that each corresponds to a dynamic cost \[88\]. Stochastic dominance is employed to define dominance relationship between random variables. Then stochastic skyline routes are the routes that are not stochastically dominated by any other routes. Given a source-destination pair, a set of stochastic
skyline routes is returned to all drivers.

In contrast, our proposal is able to return distinct personalized skyline routes for different drivers based on the drivers’ distinct driving behaviors. The existing study represents the dynamic costs of a route as the same random variables, no matter which drivers will use the route [88]. Our proposal represents the dynamic costs of a route as driver-dependent, deterministic values. Specifically, the three driving behavior modeling strategies proposed in Section 5.4.1.2 are applied to derive distinct deterministic values based on drivers’ distinct driving behaviors.

Preference learning: Preference learning [37] is being studied extensively in the machine learning community. However, although a variety of learning models and methods exist, these generally rely on manually labeled training data to enable effective learning. Rather than proposing yet another learning method, we contribute to the field by introducing a method that is able to automatically generate training data based on personalized skyline routes. In this sense, our contribution is largely orthogonal to existing proposals from the machine learning community.

We proceed to compare our work to the two most similar existing works [18, 73] that also aim to identify driving preferences. First, both existing works only support static costs, e.g., distances and numbers of traffic lights. In contrast, we support both static and dynamic costs. Since dynamic costs are time-dependent and uncertain, they are very different from static costs, which means that the existing techniques do not apply in our setting. Second, the existing works offer no support for contexts. In contrast, key contributions of our proposal are to identify multiple contexts and to obtain context-aware driving preferences. Third, one of the works [18] is only able to support two different costs, while our proposal can support an arbitrary number of costs. Fourth, the other work [73] relies on the drivers to provide manually labeled training routes (e.g., good or bad routes); instead, we utilize personalized skyline routes to automatically generate training routes. Further, a perceptron algorithm is used [73], while we propose a novel, linear optimization-based learning algorithm. Fifth, we report on a solid empirical study using a large, real trajectory data set. The methods proposed in one of the works [18] are tested on synthetic data and a small real trajectory set with 156 trajectories, without knowing the identities of the drivers. The other work [18] only reports on a case study with 20 tasks with different source-destination routes, due to the high human labor needed for labeling the training data.

5.7 Conclusion and Future Work

We study how to identify personalized, context-aware driving preferences from trajectories. We propose techniques to obtain time-dependent and uncertain dynamic travel costs from trajectories; to identify distinct contexts for drivers based on the obtained dynamic travel costs; and to identify driving preference in each context. Empirical studies with a large collection of GPS trajectories suggest that the proposed techniques are effective.

Two interesting research directions exist. First, in addition to the temporal aspect
we considered in Section 5.4.3, it is of interest to consider other aspects (e.g., spatial and spatio-temporal aspects). Second, as suggested in the empirical study, some drivers’ preferences are not fully captured (i.e., the drivers who are on the left side of the vertical line in Fig. 5.16). It is of interest to consider other forms of driving preferences, e.g., non-linear preferences, instead of the weighted-sum based linear preferences.
Bibliography


