Using GPS Data in Route Choice Analysis: Case Study in Boston

by

Anyang Hou

Bachelor of Science in Urban and Regional Planning (2008)
Beijing University, Beijing, China

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

Master of Science in Transportation

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2010

© Massachusetts Institute of Technology 2010. All rights reserved.

Author

Department of Civil and Environmental Engineering
May 20, 2010

Certified by

Emilio Frazzoli
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by

Daniele Veneziano
Chairman, Departmental Committee for Graduate Student
Using GPS Data in Route Choice Analysis: Case Study in Boston

by

Anyang Hou

Submitted to the Department of Civil and Environmental Engineering on May 20, 2010, in partial fulfillment of the requirements for the degree of Master of Science in Transportation

Abstract

The pervasive location-based technologies, such as GPS and cell phone, help us find the pattern of geographical information of human behavior and also help dig opportunities in real world. In transportation field, they help people better understand the transportation behavior and at the same time collect necessary information for us. One important aspect of its application is how people choose the route given the existing urban network. However, dealing with the excessive amount of data and the modelling of route choice behavior are two major challenges in the route choice analysis. This thesis discusses the general process in the route choice analysis, from GPS data processing, map matching to the generation of route choice sets. Besides, the Path-Size logit model is implemented to address the modelling issue. In this thesis, I develop a new effective method, which I called Point-Based Local Search Map Matching, to match the consecutive GPS data to the network data. Also, I develop a new model, which I called Random Weight Choice Set Generation Model to deal with the choice set generation problem in the route choice analysis.

The data comes from two major sources. One is the Boston car GPS data. It tells when and where a specific car is. The other is the Boston urban network data, which contains all types of roads in GIS format.

Thesis Supervisor: Emilio Frazzoli
Title: Associate Professor of Aeronautics and Astronautics
Acknowledgments

It was the best of times. It was the worst of times.
I would treat my two-year study in MIT as the best and worst of times for myself rather than for the ongoing economic crisis period. In this two-year life in the US and in my country China, I met a lot of people and things, from the world-famous professor in MIT to the legendary fund manager in China, from the righteous and smart founder of a real estate company to the ugly and cunning businessman who wanted to use a honey trap for us when we were talking business. A life filled with adventure is much colorful. Honestly, this wasn’t my previous planned destiny. It is totally unexpected. But I like it. I fit it.

First, thanks to my research supervisor Prof. Emilio Frazzoli. During the short time in your lab, I really enjoyed the research atmosphere of freedom. The research topic sheds light on many ideas of the location-based service(LBS) for me.

Thanks to Francesco Calabrese, Ketan Savla, Oleg Kozhushnyan and Amrik S. Kochhar, for guiding me and being the source of some research ideas and inspiration during the research and cooperation.

Thanks to my academic advisor Prof. Joseph M.Sussman, who was very patient in the beginning meeting of every semester. I also thanks to Patty Glidden who is the Graduate Admissions Coordinator of CEE. You gave me a lot of help during these two years.

Thanks to the China stock market. You gave me the confidence and money when I faced with the sense of frustration of being looked down upon in the end of 2008.

Thanks to the greatest time and my great country China. I realize I am in the great historical course of the China capitalism. Also thanks the book of "Das Kapital". You enlightened me to realize the consumerization of population, not the urbanization that as some economists imagined, uncovers the secret of development capitalism.

Finally, my deepest gratitude goes to my family, my parents, my little sister and my girlfriend Jie. I got courage, love and support from you when I was low in my life.
# Contents

1 Introduction .............................................. 11
   1.1 Motivations and Objectives ............................ 11
   1.2 Thesis Contribution .................................. 14
   1.3 Thesis Outline .................................... 14

2 Literature Review ...................................... 17
   2.1 Data Processing ..................................... 17
   2.2 Choice Set Generation ............................... 19
   2.3 Route Choice Model ................................ 22
      2.3.1 Shortest Path Algorithm ......................... 22
      2.3.2 Multinominal Logit .............................. 24
      2.3.3 C-Logit ......................................... 24
      2.3.4 Cross-Nested Logit .............................. 26
      2.3.5 Path Size Logit ................................ 27

3 Methodology ........................................... 31
   3.1 Trajectory Segmentation .............................. 32
   3.2 Point-Based Local Searching Map Matching ........... 35
      3.2.1 Region Classification ............................ 36
      3.2.2 Neighborhood Region Searching .................. 37
   3.3 Random Weight Choice Set Generation Model ........... 39
      3.3.1 Random Weight Choice Set Generation .......... 40
      3.3.2 Capacity of Choice Set .......................... 42
# List of Figures

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>The first situation of path extraction</td>
<td>34</td>
</tr>
<tr>
<td>3-2</td>
<td>The second situation of path extraction</td>
<td>34</td>
</tr>
<tr>
<td>3-3</td>
<td>The third situation of path extraction</td>
<td>34</td>
</tr>
<tr>
<td>3-4</td>
<td>Region Classification</td>
<td>37</td>
</tr>
<tr>
<td>3-5</td>
<td>An exceptional example of closest point in adjacent region</td>
<td>38</td>
</tr>
<tr>
<td>3-6</td>
<td>The general situation of neighborhood region of a region (x,y)</td>
<td>38</td>
</tr>
<tr>
<td>3-7</td>
<td>The upper boundary situation of neighborhood region of a region (x,y)</td>
<td>39</td>
</tr>
<tr>
<td>3-8</td>
<td>The upper-left corner situation of neighborhood region of a region (x,y)</td>
<td>39</td>
</tr>
<tr>
<td>3-9</td>
<td>The grid network</td>
<td>43</td>
</tr>
<tr>
<td>3-10</td>
<td>The situation when n=1</td>
<td>45</td>
</tr>
<tr>
<td>3-11</td>
<td>The situation when n=2</td>
<td>45</td>
</tr>
<tr>
<td>3-12</td>
<td>The situation when n=3</td>
<td>46</td>
</tr>
<tr>
<td>3-13</td>
<td>The situation when n=4</td>
<td>46</td>
</tr>
<tr>
<td>3-14</td>
<td>The Capacity when n = 20</td>
<td>49</td>
</tr>
<tr>
<td>3-15</td>
<td>The Capacity of Different Value n</td>
<td>49</td>
</tr>
<tr>
<td>3-16</td>
<td>A Simple Example of 3-by-3 Grid Network</td>
<td>50</td>
</tr>
<tr>
<td>3-17</td>
<td>The analysis of size of choice set and sampling size</td>
<td>51</td>
</tr>
<tr>
<td>3-18</td>
<td>The analysis of size of choice set and the original value of path</td>
<td>53</td>
</tr>
<tr>
<td>4-1</td>
<td>Latitude before cleaning</td>
<td>56</td>
</tr>
<tr>
<td>4-2</td>
<td>Longitude before cleaning</td>
<td>56</td>
</tr>
<tr>
<td>4-3</td>
<td>Latitude after data cleaning</td>
<td>57</td>
</tr>
<tr>
<td>4-4</td>
<td>Longitude after data cleaning</td>
<td>57</td>
</tr>
</tbody>
</table>
4-5 The research area in Boston .............................................. 57
4-6 The point and link data of Boston ...................................... 59
4-7 Analysis on the number of paths and time gap ....................... 60
4-8 A completed trajectory(1) ............................................. 60
4-9 A completed trajectory(2) ............................................. 60
4-10 An example of path(part 1) ........................................... 61
4-11 An example of path(part 2) ........................................... 61
4-12 Before map matching ................................................ 62
4-13 After map matching .................................................. 62
4-14 An Example of Revealed Path ........................................ 65
4-15 The First Example of Alternative Path ............................... 65
4-16 The Second Example of Alternative Path ............................ 65
4-17 The Third Example of Alternative Path ............................... 65
4-18 The Fourth Example of Alternative Path ............................. 65
4-19 Histogram of Distance ............................................... 66
4-20 Histogram of Link Number .......................................... 66
4-21 Data Flow of the Route Choice Analysis ............................ 68
List of Tables

4.1 GPS Data ......................................................... 56
4.2 Examples of raw data ........................................... 56
4.3 Initial Network Data .............................................. 58
4.4 Statistics on Attributes ......................................... 69
4.5 Estimation Results ............................................... 71
4.6 Model Fit Measures .............................................. 71
Chapter 1

Introduction

1.1 Motivations and Objectives

Location, location, location! The pervasive location-tagged mobile devices, such as GPS and mobile phone, greatly change our daily life and our world. These devices generate a bunch of traces left by moving objects and today they grow explosively. They not only help us identify the geographical information of objects, but also help us better understand some behaviors and activities in spatial context. Nowadays, many people develop and are still developing various kinds of location-based service (LBS). For example:

- Searching the nearest restaurant with the consideration of their preferred taste;
- Locating taxi on map and finding the nearest one for passengers;
- Location-based mobile advertising
- Tracing the entire movement in logistics area.

Even in the online social networking area which seems to have no geographical limitation, the geographical tag now plays a significantly important role in these websites, such as Foursquare, Gowalla, Loopt which are labeled as location-based networking platform.

In the transportation area, it is never too late to emphasize the importance of geographical information because the nature of travel activity is the movement from one location to another location. As a derived behavior, the combination of travel
activity correlated with other activities, such as shopping and working, can reveal some unknown human behavior patterns and afterwards develop some opportunities and strategies to improve our life. In order to achieve this goal, the first thing is to understand travel behavior first.

This thesis focuses on the route choice behavior. For an individual person, every trip of him or her indeed contains the choice of a specific path from a bunch of alternatives. He/she may consider some influencing factors in the decision-making process such as “Which route can help me save travel time?” “Should I walk on the same road to home as usual?” or “Which route is less congested during rush hour?”.

However, unlike other aspects of transportation research, typically in the mode choice, the route choice behavior faces with a large amount of alternative paths rather than a small choice set. Significantly, they are implicit and the traveler cannot list all possible choice even using the questionnaire survey.

To solve this, a series of models, which are called route choice model in general, were developed. These models can help assess traveler’s opinion of a variety of parameters of transportation network, such as distance, travel time, travel cost, the hierarchy of road. Also, the personal attributes can be considered in the route choice analysis because people have different preference which is affected by their own characteristics.

The application of route choice model can help us solve many transportation problems. For example, we can estimate the parameters of it in a specific area and then use them to develop different scenarios. We can take a look at how traffic flow changes with the growth of population. We can even evaluate what is the overall impact of traffic flow brought by a proposed construction project. Moreover, we may want to know the local impact and the overall impact with a higher charge of a toll road. Using the route choice model, we can easily analyze these kinds of scenarios and give some simulation results for further policy making.

Furthermore, the route choice model may be used in real time traffic management if the real-time traffic information is sufficient. Dynamic Traffic Management Systems improves and balances the traffic flow by providing the real-time traffic information
to travelers in a quick response. In the whole process the route choice model plays a key role because the system needs to know how travelers will react to the information which is provided by the model.

However, the difficulties of the route choice model is data processing issue, including data collection. In transportation mode choice research, it is easy to ask which mode traveler uses. On the contrary, it is hard to illustrate a detailed route of a traveler since it is involved with so many segments of transportation network. Fortunately, GPS device can record the complete trace of the travel behavior. As far as we know, an obvious drawback of GPS data is it has huge amount of data to process. Most of the time GPS data is collected for a long-enough time in a specific short-time interval form. In addition, most urban transportation networks are large-scale network which calls for a powerful computation capacity and some effective algorithms. This situation makes some obstructions in the research.

The object of this thesis is to present how to extract travel information effectively from GPS data in combination with network data in addition to the model building and estimation. A case study on the transportation network in Boston is used to illustrate the whole data processing and modeling. The object at least includes these perspectives:

- **Data Processing** Under the circumstance of the large quantity of GPS data and large-scale transportation network, an adaptive approach is needed in the map matching process which matches observed trips to the network. The processing of data, both the data structure of network and the data organization of the intermediate output should be clearly shown in a convenient way for further research as well.

- **Feasible Model** A feasible model in the choice set generation model is necessary because this affects the final result. In a common sense, a logical and feasible path choice set for a revealed path will help better evaluate the impact of parameters or even help choice the right variables. Meanwhile, the process should be operational since the number of alternatives in a large-scale network is infinite in theory.

- **Application** Within the realized data and the right procedure, a meaningful output should be expected in the end. Besides, the adaption of the method in this
thesis is case-sensitive since the research objects, city car and the city of Boston, have their own characteristics and cannot be a universal mode to be used in all situations.

1.2 Thesis Contribution

This thesis concerns the route choice analysis using GPS data of car. The main contribution of this thesis is to develop a complete process dealing with GPS data in route choice analysis, which includes:

- A new approach in map matching step which can be applied in large-scale transportation network to calibrate the GPS data with the network data;
- A new model in choice set generation process that is more effective in creating alternative paths in stochastic approach.
- A theoretical derivation for the choice set size which is on the basis of the number of links of a revealed path.
- A modification of network data which enhances the applicability of various of data format while at the same time doesn’t affect the specification and estimation of Path Size Logit model.

1.3 Thesis Outline

This thesis is structured as five chapters given in their chronological order. The remaining parts are organized as follows:

- Chapter 2 reviews the related literature. I present an overall review of choice set generation model and then give a browse of the route choice model. In this chapter, I also review some papers related to the data processing issue.
- Chapter 3 deals with the whole procedure of route choice analysis, including the trajectory segmentation from GPS data, the map matching work, the choice set generation process and the route choice models.
- Chapter 4 presents the framework for estimating route choice models using
Boston data. In this case, the detailed process from raw data to the final result is illustrated.

- Chapter 5 presents the conclusion and gives some ideas of future research perspectives.
Chapter 2

Literature Review

Route choice framework contains two main parts: choice set generation and route choice model. Comparatively, fewer research focuses on the development of realistic choice set generation. Besides, with the popularity of some data mining techniques and the difficulty in data collection, more and more attention is now on the data processing. In this part I will discuss the data processing, choice set generation and the route choice model. In section 2.1 I will review the data processing, including the traditional advantage of interview and mobile technology. In section 2.2 I will review some models on the choice set generation. In section 2.3 I will review some route choice models such as Path Size Logit (PSL) model.

2.1 Data Processing

The data processing is a critical issue in route choice analysis. The traditional and popular way is interview. Researchers can use phone interview or questionnaire or internet to get data from travelers. The advantage of this method is that the quality of data can be controlled by researchers. However, its drawback is obvious. This method is costly and time consuming.

One of the earliest paper using this method is presented by Ben-Akiva[3]. In this research he stopped cars at the road side and mailed questionnaires to car owners to collect the data. Another example is that Ramming[22] asked MIT faculties and
students to get their travel information around Boston area in his PhD dissertation. In this situation the path illustration was accurate. For some unclear details, he used the shortest path algorithm to supplement the blur information between two consecutive points. But this method may not be suitable for long distance trip. For long distance trip, it is very hard to request the traveler to give a clear statement. Vrtic[27] solved this by asking travelers to describe three intermediate cities that they passed through from the origin to the destination. Though this gives some convenience for the data collection, it causes the researcher great effort to correct missing information. However, some information still missed for the research.

The mobile technology, including GPS device and cellular phone technology, greatly helps the data collection process. This belongs to the passive approach so that researcher cannot use the stated preference in this way. The advantage of using the mobile technology in the route choice research is having large amount of revealed data which reflect the true travel behavior. However, the massive data require some advanced statistical techniques and programming background. The data accuracy is also a problem in this approach because of the technical limitation. This problem is significant when using the cellular phone data. Also, the local environment, such as tunnels and high buildings, may cause the mobile-based data inaccurate or bring a big gap in the data.

In late 1990s, Murakami[18] did an experiment on collecting travel information by using the Personal Digital Assistance(PDA) equipped with a GPS receiver. He reports that some differences exist in certain circumstance such as the longer distance, between the GPS device and the traditional data collection. Another study is that Nielsen[19] did the route choice analysis in Copenhagen with a big GPS dataset. He also proposed the map matching approach in this research. Li[16] used the disaggregated GPS data collected from 182 drivers. This paper indicated a connection between the morning route choice behavior and work schedule flexibility as well as the route attributes and traveler’s socioeconomic attributes. Rich[23] proposed a variant of the constrained enumeration method which used GPS data to build connection trees between the origin and the destination. This method was based on
the assumption that route alternatives can be formed as a set of sub-paths. Some people doubted the availability of mobile-based data. But Bar-Gera[2] proved that it was useful in various practical applications such as advanced traveler information systems and evaluating system performance for modeling and planning. He examined the performance of traffic speed and travel time with the comparison of cellular measurements and dual magnetic loop detectors.

All in all, the mobile-based approach is still a new area to explore, though it has some flaws. How to use the data becomes a specific topic for scholars since it is attractive for its low cost of data collection.

2.2 Choice Set Generation

Given an OD pair in a network, we will have a lot of paths to connect the origin and the destination point. Unlike the mode choice set which has a limited and a clear choice set, we face the difficulty in finding the possible paths in route choice analysis. Actually, there are nearly infinite choices for a driver if the network is large enough or if we allow loops in the path. Thus we need to pick up the alternative paths which should be as real as possible. The choice set generation algorithms are therefore designed to deal with this issue. A comprehensive overview of the path generation algorithm was given in Fiorenzo-Catalano’s PhD dissertation[14]. In general, the choice set generation can be classified as deterministic or stochastic approaches.

For the deterministic part, it is heavily affected by the rules that researchers set. The basic idea of this method is to set some constrains or impedances to eliminate the unrealistic routes or directly pick up the alternative paths. Generally, the more constrains there are, the less alternatives will be available with the growth of number of constrains. These various algorithms lead to many kinds of approaches to achieve this goal on the basis of shortest path or the similar idea of link cost.

One of the earliest approach is the Elimination by Aspects developed by Tversky[24]. Here some alternatives that meet the requirement will be picked up by using heuristics. For example, among all possible paths we are just interested in these alternatives
within a 1km travel distance. So that we set a threshold to let the algorithm pick those paths less than 1km. Another approach is the link elimination approach developed by Azevedo[1] which will calculate the shortest path with the generalized cost function.

The following approaches show more and more details and complicated features. One of these is the Pareto-optimal solution developed by Dell’Olmo[12]. This paper sets two phases in finding the alternative paths. In the first phase, they implement a multi-criteria shortest path algorithm to find the Pareto-optimal path. Based on this, they construct a buffer zone using the geographical information system (GIS) and find the spatially different routes, which they treat as the most feasible alternative paths.

Ben-Akiva[3] offers a new idea in this approach. He develops the labeling method to define the “labeled” paths. The idea is to transform a large number of physical routes into a smaller number of routes each representing a specific “label”. The estimated function is established to maximize the proportion of observed paths among alternative paths. This idea is widely used in this field to evaluate the performance of path generation algorithm.

Instead of the link elimination method, some researchers choose to find the alternative paths directly because they claim that the computation capacity is a limitation for the elimination method. Van der Zijpp and Fiorenzo-Catalano[25] implement the K-shortest path algorithm using the ordinary shortest path computation as its elementary operation. Another advantages of this method are its limited sensitivity to the level of restriction of constrains and the feasibility for realistically sized urban networks.

Generally, these methods generate meaningful and more heterogeneous choice sets. However, one critical issue is that they seldom consider the realistic meaning from human behavioral perspective. No matter the shortest path or the similar link penalty approach, they are attractive in computational aspect. Another weakness is that the limitation of choice set is based on the origin and destination pair and the road network rather than the diverse taste and social attributes of individuals.

For the stochastic approaches, the basic idea follows the probabilistic choice set
models suggested by Manski\cite{17} using the following expression (2.1):

\[ P_n(i) = \sum_{C \in G_n} P_n(i|C) P_n(C) \]  

(2.1)

Here:

- \( M_n \) is the set of feasible alternatives for individual \( n \) (\( M_n \in M \)) while \( M \) is the universal choice set;
- \( G_n \) is the set of non-empty subsets of \( M_n \);
- \( P_n(i) \) is the probability of individual \( n \) choosing the alternative \( i \) given the choice set \( M_n \);
- \( P_n(i|C) \) is the probability of individual \( n \) choosing alternative \( i \) given the choice set \( C \);
- \( P_n(C) \) is the probability of individual \( n \) choosing choice set \( C \) given the choice set \( M_n \).

On the basis of this idea, Ben-Akiva\cite{5} develops a stochastic method to estimate a choice set generation model by using information contained in responses to alternative availability questions. This approach incorporates the information on individual's choice set and revealed preference information.

Ramming\cite{22} generates the link cost from the probability distribution instead of a deterministic value. These link costs are introduced into the alternative paths. The shortest path is calculated on the basis of randomly distributed generalized cost. In his opinion, the generation choice set should include the revealed choice to present the unbiased choice sets.

Cascetta\cite{11} and his team members propose an operational model explicitly simulating the route perception of alternatives by drivers in the urban road network based on a sample of routes by treating the choice set as a fuzzy set. They show that only few routes are actually perceived as feasible alternatives and some attributes, such as the topological properties and the socio-economic attributes, will have an effect on the user perception.

Bovy\cite{8} developes a new stochastic approach called doubly stochastic choice set
generation model for the route choice analysis and flow prediction. This approach establishes choice set prior to the choice modeling step and set both parameters and the attributes as stochastic variables.

Frejinger[15] assumes that the choice sets contain all the paths for the OD pair although this is behaviorally questionable. This can help avoid bias in the model. They derived a sampling correction for the algorithm generating subsets of paths suitable for model estimation.

In general, the choice set generation model still keeps being improved, no matter to meet the realistic goal or the time saving goal. However, not all models find the ultimate solution. Some are easy to compute but the similarity among generated routes is the issue. Some are easy to solve the similarity issue while it is not so realistic from behavior perspectives. Besides, the limited revealed choices make these models find few suitable alternative choices since they cannot throughout a certain number of choices from the same individuals. Finally, the capacity of choice set is given subjectively. It is known that the larger capacity of the alternative choices is, the better estimation will be in the latent process. But the tradeoff of the capacity is the quality of it and the computational reality.

### 2.3 Route Choice Model

In the part, I will give an overview of route choice models about their general ideas, formulations, advantages and disadvantages. In general, these models will estimate some parameters using the revealed choice and alternative choice. These parameters represent the effect on the behavior choice by some attributes, such as distance, travel time, envision quality and so on.

#### 2.3.1 Shortest Path Algorithm

The shortest path algorithm developed by Dijkstra[13] is one of the most widely used method in many academic research. It simply searches the mimimum distance of path among all alternative paths. Its research object is a network composed of nodes and
links. Every link between two nodes has a distance value. Its main idea is described as follow:

1. Set the value of distance as zero for the origin node and infinity for other nodes.
2. Mark all nodes as unvisited label and set the origin node as current.
3. Consider all adjacent nodes as the next current label and calculate their distance from the initial node. If the distance is less than the previously recorded distance, revise the distance.
4. If all adjacent nodes of the current nodes are tried, mark them as visited label. Here the minimum distance is recorded.
5. Set the unvisited node with the minimum distance as the next current node and iterate from step 3.

The algorithm is:

```python
1  function Dijkstra(G, w, s)
2      for each vertex v in V[G]
3          d[v] := infinity
4          previous[v] := undefined
5      d[s] := 0
6      S := empty set
7      Q := set of all vertices
8      while Q is not an empty set       //loop
9          u := Extract_Min(Q)
10         S := S union {u}
11         for each edge (u,v) outgoing from u
12             if d[v] > d[u] + w(u,v)
13                 d[v] := d[u] + w(u,v)
14                 previous[v] := u
```

Apparently, this algorithm only concerns the distance, or similar link cost such as travel time, in the network. The implicit assumption is that travelers have full
knowledge about the road network while at the same time they are very logic as the “rational economic man”. However, the reality and the common knowledge show few evidences on this assumption. Nevertheless, the shortest path algorithm is a very important milestone for route choice analysis and many researches are based on it.

2.3.2 Multinominal Logit

The basic idea of Multinominal Logit (MNL) model is to assume individuals have some errors in the utility term. These errors are the same for every individual and have a specific type of distribution, normally the Gumbel distribution. The model is written as:

\[ P_i = \frac{e^{-\theta L_i}}{\sum_{j \in C_n} e^{-\theta L_j}} \quad (2.2) \]

Here:

- \( P(i) \): the probability of an individual choosing path \( i \);
- \( \theta \): the coefficient of utility;
- \( L_i \): the length of path \( i \);
- \( C_n \): the choice set of alternative path for individual \( n \).

Obviously, it calls for the revealed paths and alternative paths with their attributes such as road type, distance, travel time and so on. The MNL model in route choice is not different as other choice models except that in route choice the amount of alternative choices is larger than in other choice models.

2.3.3 C-Logit

Cascetta[10] introduced a correction term called commonality factor (CF) into the utilities in order to overcome the path overlapping problem of MNL model. Its expression is:

\[ P(i|C_n) = \frac{exp(V_{in} + CF_{in})}{\sum_{j \in C_n} exp(V_{jn} + CF_{jn})} \quad (2.3) \]
Here, $V_{in}$ and $V_{jn}$ are the utility functions of route $i$ and $j$ for individual $n$. The commonality factors $CF_{in}$ and $CF_{jn}$ are the measurement of similarity of route $i$ and $j$ for individual $n$. The value of $CF$ is always negative since overlapping paths will have lower percentage as expected. Cascetta gave four different expressions for the commonality factor corrections [10, 9]:

\[
CF_{in} = -\beta_0 \ln \sum_{j \in C_n} \left( \frac{L_{ij}}{\sqrt{L_i L_j}} \right)^\gamma
\]  
(2.4)

\[
CF_{in} = -\beta_0 \ln \sum_{a \in \Gamma_i} \left( \frac{l_a}{L_i} \sum_{j \in C_n} \delta_{aj} \right)
\]  
(2.5)

\[
CF_{in} = -\beta_0 \sum_{a \in \Gamma_i} \left( \frac{l_a}{L_i} \ln \sum_{j \in C_n} \delta_{aj} \right)
\]  
(2.6)

\[
CF_{in} = -\beta_0 \ln \left[ 1 + \sum_{j \in C_n, j \neq i} \left( \frac{L_{ij}}{\sqrt{L_i L_j}} \right) \left( \frac{L_i - L_{ij}}{L_j - L_{ij}} \right) \right]
\]  
(2.7)

Here:

$CF_{in}$: the commonality factor value of route $i$ for individual $n$.

$L_i$: the length of route $i$.

$L_{ij}$: the shared length of route $i$ and route $j$.

$l_a$: the length of link $a$.

$\delta_{aj}$: dummy variable, which is 1 if the link $a$ is part of route $j$ and 0 otherwise.

$\Gamma_i$: the set of links in route $i$.

$\beta_0$: parameter to be estimated.

$\gamma$: parameter to be estimated.

From these formulations, we can conclude that the formulation (2.4) emphasizes the shares of links between route $i$ and other routes. This formulation tries to find the common length among routes. The formulation (2.5) and the formulation (2.6) emphasizes the relationship between the route $i$ and all its links. Unlike the formula-
tion (2.4), these two formulations try to look into the link-based structure of route and see how many routes use these links. The differences between formulation (2.5) and expression (2.6) is the value of weights on the link. The last formulation (2.7) emphasizes the similarity between of routes while at the same time it considers the non-shared parts as an important factor.

The C-logit model solves the overlapping problem. But the subjective formulation to define the commonality is a disadvantage of it. It is hard to say which formulation among these four choices is the best formulation. Besides, an implicit assumption of these formulations is that the length is the only factor to be considered among routes. However, we may expect that some aspects, such as the similarity of visions and flows should have some effect on the route choice behavior. These are unseen in the model.

2.3.4 Cross-Nested Logit

The Cross Nested Logit (CNL) model is proposed by Vovsha[26]. On the basis of it, Prashker and Bekhor[21] gives the mathematical formulation of this model. The basic idea of this model is that some choices have common points so that they are chosen within nests. The formulation of this model is simple:

\[ P_k = \sum_m P(m) P(k|m) \]  \hspace{1cm} (2.8)

Here:

\( P(m) \): the marginal probability of choosing a nest m;

\( P(k|m) \): the conditional probability of choosing route k in nest m.

These two parts, the marginal probability and the conditional probability, can be calculated using these two formulas:

\[ P(m) = \frac{\left( \sum_k (\alpha_m k \exp(V_k))^{1/\mu_m} \right)^{\mu_m}}{\sum_h \left( \sum_k (\alpha_h k \exp(V_h))^{1/\mu_m} \right)^{\mu_m}} \]  \hspace{1cm} (2.9)
\[ P(k|m) = \frac{(\alpha_{mk}\exp(V_k))^{1/\mu_m}}{\sum_l (\alpha_{ml}\exp(V_l))^{1/\mu_m}} \] (2.10)

Here:
\( \alpha_{mk} \): the inclusion coefficients where \( 0 \leq \alpha_{mk} \leq 1 \) and \( \sum_m \alpha_{mk} = 1 \);
\( \mu_m \): the nesting coefficients where \( 0 \leq \mu_m \leq 1 \).

A further relationship between the inclusion coefficient and the link in a route is[20]:

\[ \alpha_{mk} = \frac{L_m}{L_k} \delta_{mk} \] (2.11)

Here:
\( L_m \): the length of link \( m \);
\( L_k \): the length of route \( k \);
\( \delta_{mk} \): the dummy variable, equals to 1 if route \( k \) contains the link \( m \) and 0 otherwise.

This model is logical for solving the link sharing among different routes. However, it is not appropriate to use it in large-scale road network since any link in the network will be shared among large amounts of routes. It will need a long computation time to estimate the model. Also, Ramming[22] pointed out that this model tends to collapse to MNL model and has no advantage comparing with MNL model.

### 2.3.5 Path Size Logit

The Path Size Logit(PSL) model is developed by Ben-Akiva and Ramming[6]. Another paper from Ben-Akiva and Bierlaire[4] also presents it. This model is for an application of discrete choice theory for aggregate alternatives. Similar to C-Logit, the PSL model tries to solve the path overlapping problem by adding a correction term to the utility function:

\[ P(i|C_n) = \frac{\exp(V_{in} + \ln PS_{in})}{\sum_{j \in C_n} \exp(V_{jn} + \ln PS_{jn})} \] (2.12)
Here:

\( P(i|C_n) \): the probability of path \( i \) chosed by individual \( n \) in choice set \( C_n \).

\( V_{in} \): the utility of path \( i \).

\( PS_{in} \): the path size factor of path \( i \).

Unlike the C-Logit model, the "path size" factor in PSL model indicates that the proportion of the path that constitutes a completed path. But in the C-Logit model the commonality factor reduces the utility of a path. If a path has no overlapping with other paths, it needs no adjustment in the utility part and thus the value of \( PS_{in} \) equals to 1 and the value of \( lnPS_{in} \) is zero.

The correction term \( PS_{in} \) also has different forms as the C-Logit:

\[
PS_{in} = \sum_{a \in \Gamma_i} \frac{L_a}{L_i} \frac{1}{\sum_{j \in C_n} \delta_{aj}}
\]

\[
PS_{in} = \sum_{a \in \Gamma_i} \frac{L_a}{L_i} \frac{1}{\sum_{j \in C_n} \frac{L_{C_n}}{L_j} \delta_{aj}}
\]

\[
PS_{in} = \sum_{a \in \Gamma_i} \frac{L_a}{L_i} \frac{1}{\sum_{j \in C_n} \left( \frac{L_a}{L_j} \right)^\varphi \delta_{aj}}
\]

\[
PSC_{in} = \sum_{a \in \Gamma_i} \frac{L_a}{L_i} ln\left( \frac{1}{\sum_{j \in C_n} \delta_{aj}} \right)
\]

Here:

\( PS_{in} \): the path size of route \( i \) for individual \( n \).

\( L_a \): the length of link \( a \).

\( L_i \): the length of route \( i \).

\( C_n \): the choice set for individual \( i \).

\( L_{C_n} \): the length of the shortest path in choice set

\( \delta_{aj} \): dummy variable, equals 1 if the link \( a \) is a part of route \( j \) and 0 otherwise.

The expression (2.13) is the original form that comes from the paper of Ben-Akiva.
and Ramming[6]. It considers the weight of each link as the proportion of length in the route while at the same time it considers the number of routes that use a specific link. Obviously, if \( n \) routes use it, its value is \( 1/n \). The second expression (2.14) is introduced by Ben-Akiva and Bierlaire[4]. Apparently, the relationship between shortest path and alternative paths is indicated by this formulation. Ramming introduces the third expression (2.15). In this formulation he adds a parameter \( \varphi \). When \( \varphi \) equals to zero, this formulation turns to the formulation (2.13). This formulation is for the consideration of the influence of excessive long routes on the utility of shorter paths[22]. This formulation is the generalized form of path size term. The last formulation (2.16) is developed from random utility theory based on aggregate alternatives and GEV models[7]. In this expression the logarithm is included in the sum term instead.
Chapter 3

Methodology

In this chapter, I present an overall methodology that can be used to model travel behavior using the raw GPS data. In general, the required input of a route choice model includes the revealed trips and the alternative choice. The first challenge of using GPS data is that it cannot distinguish the origin and the destination point for each trip because all data are consecutive with time and space information. It is just the trajectory for a traveler within a period. Therefore, the first step is to extract the Origin-Destination points and the corresponding paths.

After this step, the revealed paths are available. The second step is to match these paths to the road network data. The revealed paths are still illustrated by a sequence of points while the road network data is illustrated by nodes and links. So in this step it is necessary to match the revealed path in the same form of road network data. This also can help calculate some characteristics of revealed path, such as distance, link number, road type and so on. Also, the precision issue of GPS device also calls for this step. Not all points generated by GPS will be absolutely accurate. The map matching work thus needs to calibrate them and make sure there are few unacceptable outliers.

The third step is the choice set generation process. After two above steps, we will have a standard data structure similar to the network data for revealed paths. Hence it should find the corresponding choice set for each revealed trip. The choice set should be realistic enough to represent the possible solution for a traveler from
one place to another. It should be also be operational because most of the time the number of alternative paths is a large number, sometimes infinite. In this part I create a new method which is called random weight choice set generation, to find the right number of alternative paths in effective way. This method can be used in large-scale network which is seldom seen today. The output of this step is a large number of alternative paths with some variables that illustrate their characteristics.

The fourth step is the model specification and model estimation. In this part I use the Path Size Logit(PSL) model to estimate the result.

3.1 Trajectory Segmentation

For a specific trip, the origin point and the destination point is clear. For example, in a trip from home to the office home is the origin point and the office is the destination point. In a questionnaire survey the OD issue is not a problem because most of the time participants are asked to state the OD pair clearly. However, there is no label to declare which point is origin and which point is destination in the GPS data. Some researches using the GPS data also pointed out this problem. This problem varies among different transportation mode. For buses, their repeated routes may help them distinguish the trip and even help judge the location of bustop though there is no available network data for them. But in our case, the movement of car is not so regular, which calls for a specific method to deal with it.

Fortunately, the working mechanism of the GPS device provides the possibility to solve this problem. If the time gap between two consecutive points exceeds a given threshold, such as 5 minutes, it is likely that the earlier point is the destination of a trip and the latter point is the origin point of another trip.

The origin form of trajectory is:

\[ Trajectory(CarID) = \langle point_1, point_2, ..., point_n \rangle \]  

(3.1)

Each point contains the location and time information. The location is represented
as latitude and longitude. The time is recorded as the form of UNIX time.

So the method to create trips from trajectories is to set an appropriate threshold and use it to separate the whole trajectory to several paths. The data structure of outcome is similar to that of the trajectory:

\[
Path(PathID, CarID) = \langle point_1, point_2, ..., point_n \rangle
\]  \hspace{1cm} (3.2)

On the other hand, a question should be raised that, can we divide a completed path into several paths as what we do for the trajectory? This is an issue because we cannot guarantee a perfect outcome that accurately represents every real trip from the data. Some paths may be divided into two or even more paths because of the unexpected reason. In common sense, a traveler will have different actions when he/she faces with different kinds of situations. He/she may have enough knowledge to pick up the shortest path near home but he cannot do the same thing from downtown to a outlet in far distance. However, when the traveler is in a specific region, no matter he is from outside to somewhere of this region, or he is from one place in the region to the other place in the region, he will use his knowledge about this region to help him in the decision-making process. Many researches points out that a large proportion of trips are multi-purpose which may contain different activities in a single trip. For example, from the office to home, a woman may firstly pick up her children from school, then go shopping with her child, and finally go back to home from shopping center. In this situation, a path may appear to be very strange. It may contain some loops, or it may be very costly. So from this perspective, we cannot treat a single path as several shorter paths grouped together if our object is the path itself. But if our object is the reaction of a traveler in a specific region, we can treat the part which is located in this smaller region from a completed path which should be observed in a broader geographical context.

There are different situations to extract the in-region paths from the overall paths. The first situation is that all paths are in the region before extraction. In this case, the outcome is the same as income (figure 3-1).
The second situation is that part of the path is in the region and the rest are out of the region while the whole path through the region only once. In this situation we will “clip” the in-region path from the completed path. Obviously, the origin point and/or the destination point may be in the boundary of the region (figure 3-2).

The third situation is that part of the path is in the region and the rest are out of region but the whole path through the region twice. In this situation the “clip” result will be two paths (figure 3-2). Theoretically there are infinite situations considering the path can go through the region in many times. But in my research I don’t find any situation that the path goes across the region for more than 2 times.

In our case, I limit the research area in downtown Boston plus a part of Cambridge. One reason is that the overall geographical area that is covered by cars is too large. Some points are even located outside Massachusetts, such as Connecticut and New Hampshire. It is not necessary to consider all points from the data. The other reason...
is that we want to do some comparative study between cell phone data and GPS data in the future. Since the cell phone data is located in the downtown Boston, we need to limit the GPS data in the region to match the cell phone data. Importantly, we are mainly interested in the traffic situation in congested area, which is typical in downtown. Those trips that use interstate highway should be different from the trips in downtown even they are in the same mode. More details will be introduced in Chapter 4.

3.2 Point-Based Local Searching Map Matching

The purpose of this part is to match revealed path, generated from previous step, to the network data.

The format of network data in my research is the shapefile of GIS data. To have better results from the Matlab, we need to transfer it in the graph structure:

\[
Graph = \langle \text{Node}, \text{Link} \rangle
\]  

(3.3)

The node and link are expressed in this form:

\[
\text{Node}(\text{NodeID}) = \langle \text{latitude}, \text{longitude} \rangle
\]  

(3.4)

\[
\text{Link}(\text{LinkID}) = \langle \text{Node}_i, \text{Node}_j \rangle
\]  

(3.5)

The node i and node j are the end points of the link. Due to the data limitation, all links are the undirected links. In a word, the graph is an undirected graph.

So the outcome of map matching process using the generated paths is:

\[
\text{Path}(\text{PathID}, \text{CarID}) = \langle \text{Node}_1, \text{Node}_2, ..., \text{Node}_n \rangle
\]  

(3.6)

The general idea to transform the "point" to the "node" is to find the closest node of the network for each GPS point. From the geometry knowledge we know that given two points with the latitude(lat) and longitude(long) separately, we have
this model to calculate the distance between them:

\[
\text{Distance}(\text{Node}_1, \text{Node}_2) = R \arccos \left[ \cos(\text{lat}_1) \cos(\text{lat}_2) \cos(\text{long}_1 - \text{long}_2) + \sin(\text{lat}_1) \sin(\text{lat}_2) \right]
\]

(3.7)

Using this model we can match all GPS points to the road network on the basis of point-based method.

An alternative solution is to find the closest line of the network for each GPS point and then transform them to the point located in the line. In this research, this method is not appropriate because of the data structure of network which is illustrated in points. If we want to find the closest line for a specific point, we need to formulate the line at first. In this case, the computation time will become a significant issue.

The point-based method seems to be logical while it also has the computation problem. If we need to match \( n \) GPS points into the network which contains \( m \) nodes, we need to compute \( n \cdot m \) times of distance. In a large-scale network, it may be up to billions of times in finding the right solution. To solve this problem, we develop a method called Point-based Local Searching Map Matching (PLSMM).

### 3.2.1 Region Classification

The general idea of PLSMM method is to classify all points and nodes into different regions. The only searching area for a specific GPS point is to search the nodes in the same region. So before matching work we need to label all points and nodes with consideration of their geographical information. Here are some notations:

- \( l_1 \) = maximum logitude of region;
- \( l_2 \) = minimum logitude of region;
- \( w_1 \) = maximum latitude of region;
- \( w_2 \) = minimum latitude of region;

If we want to divide the whole region into \( i \times j \) small regions, we will have the
length and the width of each small region as:

\[
L = \frac{l_1 - l_2}{i} \quad (3.8)
\]

\[
W = \frac{w_1 - w_2}{j} \quad (3.9)
\]

If a point \( A \) with the latitude \( \text{lat}_A \) and the longitude \( \text{long}_A \) satisfies the following formulas:

\[
L \cdot (x - 1) \leq \text{long}_A < L \cdot x \quad (3.10)
\]

\[
W \cdot (y - 1) \leq \text{lat}_A < W \cdot y \quad (3.11)
\]

We can label it to the region\((x,y)\). For all points, we repeat this process and then get all regions labeled.

### 3.2.2 Neighborhood Region Searching

However, there is the possibility that the closest node to a point is not in the region but in the adjacent region. This may happen when a point is near the boundary of the region and the distribution of points is sparse in its region (figure 3-5). Another possibility is that there is no node in the same region for the point. In case of these situation, we need to revise the model to fix them. The general idea is to search and calculate the distance between the point and nodes which are in neighborhood
regions. The example in the figure shows that point C is closer to point A than point B although point B and point A are in the same region.

The general situation for the neighborhood regions of a region \((x,y)\) is the surrounding regions, which are \((x - 1, y - 1), (x, y - 1), (x + 1, y - 1), (x - 1, y), (x + 1, y), (x - 1, y + 1), (x, y + 1)\) and \((x + 1, y + 1)\).

If the region in which the point is located is near the boundary of the research area, it has only 5 neighborhood regions as the figure shows. The neighborhood regions vary and are decided by which boundary of the area, the upper one, lower one, left one or right one, it is next to. For example, if the region of the point is next to the upper boundary of the research area, its neighborhood regions are \((x - 1, y), (x + 1, y), (x - 1, y + 1), (x, y + 1)\) and \((x + 1, y + 1)\).

If the region in which the point is located in the corner of the research area, it has only 3 neighborhood regions. In this situation, there are four possibilities of the
neighborhood region because it may be located in the one of four corners of the area. For example, if the point region is in the upper-left corner, its neighborhood regions are \((x, y + 1), (x + 1, y)\) and \((x + 1, y + 1)\).

After the local searching improvement, the efficiency of the computation is much better than the one without improvement. In my case the previous times of computation is about 3 billion times while the improved one is only 270 million times, less than 1 percent of the previous one, with the trade off of very limited increasing of capacity of the data.

### 3.3 Random Weight Choice Set Generation Model

In general, the principle of the choice set generation model is to generate several paths from the OD pair. Since these paths are not real, two questions should naturally be raised. The first one is whether these paths are logical and realistic from behavioral perspective or not? The second one is how many paths should be generated for a specific OD pair?

Admittedly, the choice set generation is still a problematic issue in transporta-
tion research. As mentioned before, it has the large implicit amounts of alternative choices comparing with other travel choice, for example, mode choice. This not only causes the processing and visualization problem but also affects the model estimation process. Typically, the capacity of choice set has a deep impact on the result.

In this part, I will illustrate the general idea of the so-called Random Weight Choice Set Generation Model first. Then I will conduct the suitable capacity of the choice set which depends on the number of links from the origin point to the destination point.

### 3.3.1 Random Weight Choice Set Generation

Generally, there are four types of choice set generation methods. They are deterministic shortest path-based methods, stochastic shortest path-based methods, constrained enumeration methods and the probabilistic methods. The Random Weight Choice Set Generation (RWCS) model should be categorized as the stochastic shortest path-based method.

The idea of RWCS method is to treat the network traffic as a dynamic network. As we know the traffic situation is always changing in different times. The most obvious phenomenon is the rush hour in the morning and in the afternoon due to the home-to-office trip. In this case people may have different expectation on the traffic situation and thus choose different paths which seem to minimize their cost or maximize their utility.

The variance of urban network includes not only the traffic situation but also many aspects, such as the travel speed, safety measurement and the environmental quality. All these variables vary from period to period. For example, we may find that the traffic flow of a specific link is normal distributed. Most of the time a traveler may choose this link because it has less traffic flow. But sometimes it is so heavily congested that the traveler choose another route to maximize his utility. So for each link \( i \) for individual \( n \) we may have this generalized cost function:

\[
C_{in} = w_1 X_{i1} + w_2 X_{i2} + \ldots + w_k X_{ik}
\]

(3.12)
Notation:

- $C_{in}$: the generalized cost of link $i$ for individual $n$;
- $X_{ink}$: attributes $k$ for link $i$ such as traffic flow;
- $w_k$: the random weight for variables $X_{ink}$.

In principle, the distribution of random weight for different links and different attribute of it should be various. This illustrates the idea that each attribute of each link should have its own characteristics, such as mean value and variance. To simplify this problem and to make sure there is no negative value of the attributes, we suggest use this form to calculate the random weight:

$$w_k = e^{N(\mu_k, \sigma_k^2)}$$  \hspace{1cm} (3.13)

In principle, this formulation should not be the unique solution for different places. For instance, if we observe the past performance of each link and know all mean value and variance of each link in a city, we can easily calibrate it or write a new one with different types of distribution.

After having the generalized cost of each link, we can calculate the total cost ($PC_{in}$) of the path $i$ for individual $n$:

$$PC_{in} = \sum_{a \in \Gamma_i} C_{an}$$  \hspace{1cm} (3.14)

It is very simple since the cost of path $i$ is the sum of cost of its all links. So the selected path is the minimum-cost path among all possible path $i$. This can be calculated using the Dijkstra algorithm. If we repeat the above process, which should include the random weight assignment and the shortest path calculation, in $m$ times, then we will find $m$ paths which are all "minimum cost" paths at each time. Notice the value of $m$ is the capacity of choice set. I will give a detailed discussion on the capacity of choice size issue later. Assume these paths are labeled as path 1, path 2, ..., path $m$ for a specific OD pair, the choice set is:

$$ChoiceSet = \{Path1, Path2...Pathm\}$$  \hspace{1cm} (3.15)
Notice, some paths may be repeated because it is possible that the minimum paths of different computational times are the same. So the real final capacity of choice set will be equal to or smaller than $m$.

There are some characteristics of this model. First, the outcome of model, which is the path with minimum cost, is affected by the diversity of the cost of links. An extreme example is that there are two paths which only contain one link. The origin cost of each link is 1 and 100 separately. Though it will multiple their weight later, it is very unlikely that the minimum path will be the one with origin cost of 100. Second, the result will be affected by the variance of weight for each link, which is understandable.

A further exploration of this model would be the relationship of weight among different links. In fact, there is an implicit assumption in the model. It assumes that every link is independent and has no affect on other links. But in a real network this is not true. If a trunk road is heavily congested, its adjacent roads will have a higher probability of heavy traffic flow because the traffic flow of the trunk road will be forced to choose adjacent roads. Even though this is a worthy discussion, I will not give the deep discussion on it due to the data limitation.

### 3.3.2 Capacity of Choice Set

The capacity of choice set is a critical issue in choice set generation problem. Most of the time the majority of alternative paths are not observable in the research, this situation calls for the appropriate number of choices. Empirically, the more complex and the larger urban road network is, the more availability for route choice will be.

A direct observation may conclude that the number of alternative route should relate to the distance between the origin point and the destination point. In a word, the capacity of choice set is a function of distance. In a common situation it is roughly right because when the path becomes longer, the urban network appears to be much larger and complicated. However, we know that a network is composed of links and nodes. They are the basic elements that build the network with not only the distance attributes but also the connection among elements. The connection is irrelevant to
the physical distance. An extreme example is that a line with two nodes at each end of this line is the only one alternative path to connect these two nodes, no matter how long the line is. On the contrary, we may face a huge amount of alternative choice if we are in a small town with a bunch of walking paths, though the distance among two arbitrary points in this town is quite short.

This enlightens us that the capacity of choice set maybe a function of the number of links. But a question is raised: what is the relationship, typically in theory, between them? To solve this, we need to imagine an ideal road network.

The road network is a grid network with equal links. First, we need to inform that the morphology of a network is an important attribute for it. A grid network and a radiated network have different performance even if their other attributes are exactly the same. A hybrid road with various forms will show much complicated situations. In order to simplify this problem, together with the consideration of the popularity of grid network, we assume this ideal network as the grid form. Second, the equal length of links assumption helps to exam how the connections affect the capacity of choice set. A similar concept is the Manhattan distance, which considers the right angle length instead of direct distance. Here we need to define:

\[ n(a, b) \]: there are \( n \) links which are composed of \( a \) links in the W-E direction and \( b \) links in the N-S direction from the origin point to the destination point. Apparently, \( n = a + b \) for the grid network. For example, the 7(3,4) represents that there are 7 links to connect the origin point and the destination point in equal distance (and also
the shortest distance). The elements of this 7 links are 3 W-E links and 4 N-S links. In a grid network, \( n \) links can be expressed as:

\[
n = 0 + n = 1 + (n - 1) = 2 + (n - 2) = 3 + (n - 3) = \ldots = (n - 2) + 1 = (n - 1) + 1 = n + 0
\]

These simple but various equations represent different structures of network for the same amount of links in a path. For example, the \( 1 + (n - 1) \) means the W-E distance of the two points is exactly 1 link while the N-S distance of the two points is \( n - 1 \) links. However, the \( (n - 1) + 1 \) means the W-E distance of the two points is exactly \( n - 1 \) links while the N-S distance is 1 link. However, the form of these two types is the same because here we just concern the alternative choices for each form and they are the same for the symmetric form. In particular, if \( n \) is an odd number, we have \( \frac{n+1}{2} \) types of form because of the symmetry. If \( n \) is an even number, we have \( \frac{n}{2} + 1 \) types of form.

However, given a specific number of links, we do not know the exact structure of network. We do know that different structure of network may have different number of alternative choices even if their numbers of links are the same. Later we will prove that the smaller difference between \( a \) and \( b \) is the larger capacity of choice set for a given specific number of link will be. Now we define the number of alternative choice for a given specific number of link as:

\[
f(n) = \max n(i, j), \text{ here } i+j= n
\]

Here we illustrate the problem from the simplest example to the general pattern. To understand better, we define the coordinate for the origin point as \( O(0, 0) \) in the figure. Thus, the coordinate for the destination point is \( D(a, b) \). Also, the coordinate for other points besides the origin and the destination points follow the same idea.

When \( n = 1 \):

In this situation we have only one route. That is \((0, 0)\rightarrow (0, 1)\). Though we have another possibility which the coordinate of destination point is \( D(1, 0) \) instead
of \( D(0,1) \), the structure is the same as before.

So: \( f(1) = 1(1,0) = 1 \)

When \( n = 2 \):

In this situation we need to pass through two links to arrive at the destination point from the origin point. One possible form is the coordinate of destination point is \( D(2,0) \) and the other possible form is \( D(1,1) \).

For the \( D(2,0) \), there is the only one route:

1. \((0,0) \rightarrow (1,0) \rightarrow (2,0)\).

For the \( D(2,0) \), there are two possible routes:

1. \((0,0) \rightarrow (0,1) \rightarrow (1,1)\);
2. \((0,0) \rightarrow (1,0) \rightarrow (1,1)\).

Notice, we can reach to the destination point by many other possible routes if we allow loop in the route. For example: \((0,0) \rightarrow (1,0) \rightarrow (2,0) \rightarrow (2,1) \rightarrow (1,1)\)

But in this problem we want to find the optimal possible capacity of choice set, we do not consider this kind of situation.

So: \( f(2) = 2(1,1) = 2 \).

When \( n = 3 \):
In this situation we have 3 links in our path. The number of forms that we consider is $\frac{3+1}{2} = 2$. They are: $D(3, 0), D(2, 1)$.

For the $D(3, 0)$, we can arrive at the destination point through route:
1. $$(0, 0) \rightarrow (1, 0) \rightarrow (2, 0) \rightarrow (3, 0).$$

For the $D(3, 0)$, we have three routes:
1. $$(0, 0) \rightarrow (1, 0) \rightarrow (2, 0) \rightarrow (2, 1);$$
2. $$(0, 0) \rightarrow (1, 0) \rightarrow (1, 1) \rightarrow (2, 1);$$
3. $$(0, 0) \rightarrow (0, 1) \rightarrow (1, 1) \rightarrow (2, 1).$$
So: $f(3) = 3(1, 2) = 3.$

When $n = 4$:

In this situation we have 4 links in our path. The number of forms that we consider is $\frac{4+1}{2} = 3$. They are: $D(4, 0), D(3, 1), D(2, 2)$.

For the $D(4, 0)$, we can arrive at the destination point through route:
1. $$(0, 0) \rightarrow (1, 0) \rightarrow (2, 0) \rightarrow (3, 0) \rightarrow (4, 0).$$

For the $D(3, 1)$, we have four routes:
1. \((0,0) \to (1,0) \to (2,0) \to (3,0) \to (3,1)\);
2. \((0,0) \to (1,0) \to (2,0) \to (2,1) \to (3,1)\);
3. \((0,0) \to (1,0) \to (1,1) \to (2,1) \to (3,1)\);
4. \((0,0) \to (0,1) \to (1,1) \to (2,1) \to (3,1)\).

For the \(D(2,2)\), it is different from the previous enumeration. In this case we have 6 routes:
1. \((0,0) \to (1,0) \to (2,0) \to (2,1) \to (2,2)\);
2. \((0,0) \to (1,0) \to (1,1) \to (2,1) \to (2,2)\);
3. \((0,0) \to (1,0) \to (1,1) \to (1,2) \to (2,2)\);
4. \((0,0) \to (0,1) \to (0,2) \to (1,2) \to (2,2)\);
5. \((0,0) \to (0,1) \to (1,1) \to (1,2) \to (2,2)\);
6. \((0,0) \to (0,1) \to (1,1) \to (2,1) \to (2,2)\);

So: \(f(4) = 4(2,2) = 6\).

Now the question becomes. What the theoretical capacity for a generalized situation \(n(a,b)\) is? It is expected that the capacity will grow exponentially with the growth of number of link \(n\). Here the recursive idea is helpful to solve it. The first step for the generalized situation \(n(a,b)\) has two possibilities. The first one is \((0,0) \to (1,0)\). After this step, an individual faces the rest situation for the \(n-1\) step. Now we can reset the origin point as \(O(1,0)\) compared with the previous \(O(0,0)\) while at the same time the destination point is still \(D(a,b)\). The second one is \((0,0) \to (0,1)\). After this step, the individual faces the rest situation for the \(n-1\) step. Now we can reset the origin point as \(O(0,1)\) comparing with the previous \(O(0,0)\) while at the same time the destination point is still \(D(a,b)\). So the solution for \(O(0,0) \to D(a,b)\) can be transferred to the sum of these two situation:
1. \((0,0) \to ((1,0) \to ... \to (a,b))\);
2. \((0,0) \to ((0,1) \to ... \to (a,b))\).

From this idea we know that if we know the solution for the package of the \((1,0) \to ... \to (a,b)\) and \((0,1) \to ... \to (a,b)\), we can add them up and get the
solution for $O(0, 0) \rightarrow D(a, b)$.

A specific situation must be pointed out. How about the solution for $O(0, 0) \rightarrow k(i, 1)$ (or $O(0, 0) \rightarrow k(1, i)$, here $k=i+1$)? It is easy to find out there are $k$ routes for it.

1. $(0, 0) \rightarrow (1, 0) \rightarrow \ldots \rightarrow (i, 0) \rightarrow (i, 1);
2. (0, 0) \rightarrow (1, 0) \rightarrow \ldots \rightarrow (i-1, 0) \rightarrow (i-1, 1) \rightarrow (i, 1);
\vdots
k. (0, 0) \rightarrow (0, 1) \rightarrow (1, 1) \rightarrow (2, 1) \rightarrow \ldots \rightarrow (i, 1).

Now the solution for a generalized situation $n(a, b)$ is to split it into the basic situation $k(i, 1)$ and sum them up. The algorithm for this solution is:

function split(a,b)
if (a == 1 \| b == 1)
    k = a + b;
else
    a = split(a-1,b) + split(a,b-1);

As mentioned before, for the given number $n$ of links, there are $\frac{n+1}{2}$ types of form if $n$ is an odd number, or $\frac{n}{2} + 1$ types of form if $n$ is an even number. The difference between $a$ and $b$ in the $n(a,b)$ relates to the number of alternative choices. This result can be seen from the split algorithm. Since the algorithm will generate 2 parts of $n - 1$ links for the $n$ links, the small difference of $a$ and $b$ will lead to a higher power for it. Thus,

if $n$ is an even number:

\[ f(n) = \max \ n(i, j) = n(\frac{n}{2}, \frac{n}{2}) \]

if $n$ is an odd number:
Figure 3-14: The Capacity when $n = 20$

![Graph showing the capacity when n = 20]

Figure 3-15: The Capacity of Different Value $n$

$$f(n) = \max n(i, j) = n\left(\frac{n+1}{2}, \frac{n-1}{2}\right)$$

Here is the example (figure 3-14) that shows different combination of $i$ and $j$ when $n = 20$.

The figure 3-14 shows the capacity of choice set for different value of the number of links.

The capacity increases exponentially with the increasing of links. From the figure we can see that when $n = 20$ the capacity is up to 184756. It is easily to conclude that the capacity of choice set for most O-D pairs in large-scale urban network is huge. The question here is whether the huge number is appropriate for human choice behavior or not. Intuitively, people have limited choices when facing the route choice
from one place to the other place. But if the distance between two places is close enough, or if people are much familiar with the environment, their choices maybe close to an optimal solution. But when in a large urban network, people may just know a few of alternatives and pick one that seems to be optimal for them. In this case, the upper bound of capacity, no more than 100 for instance, is necessary.

3.3.3 Illustrative Example

Here I use a simple example to illustrate the random weight method. This is a simple grid network which looks like a 3-by-3 matrix, or say, looks like a Sudoku game(figure 3-16). Here are some notations:

\( N(a, b) \): Node of the network, for instance, \( N(2, 3) \);

\( N(i_a, i_b) \rightarrow N(j_a, j_b) \): Link with the node \( i \) and node \( j \) as the end point of it, for instance, \( N(2, 3) \rightarrow N(3, 3) \).

Apparently, this simple network has 16 nodes and 24 links. We set the origin point and destination point as \( N(0, 0) \) and \( N(3, 3) \) respectively. From the previous discussion we know that there are 20 paths from the origin point to the destination point. In other words, the full size of this OD pair is 20. Each paths contains 6 links.

Here I set the cost of all 24 links as the value of 1, except these 5 links, \( N(0, 0) \rightarrow N(1, 0), N(0, 1) \rightarrow N(1, 1), N(1, 1) \rightarrow N(1, 2), N(1, 2) \rightarrow N(2, 2), N(2, 2) \rightarrow N(2, 3) \), as the value of \( x \). The default value of \( x \) is 0.9. The purpose of setting
value is to make sure there is a minimum cost path from the origin to the destination.

Now we use the Monte-Carlo Simulation technique to do the experiment. We will generate 24 random weights which have the same mean value 0 and the variance 1. The final weight of each link equals to the random weight multiplied by the origin weight. The random weight is calculated using the formula $e^{N(0,1)}$. They are all normal distributed.

- Analysis 1: What is the relationship between the size of choice set and the sampling times? In this analysis, we have:
  
  Full size of path: 20
  Experiment times: 1000
  Sampling size: x
  Cost of link: 1 for 19 links and 0.9 for other 5 links mentioned above

  For each experiment, we will generate x paths as the choice set using the random weight method. The result shows that the number of paths is a concave curve. When the sampling size equals to the full size of path, the average number of paths is about

Figure 3-17: The analysis of size of choice set and sampling size
13. This means that the Random Weight Choice Set Generation Model will have little chance in generating a full size choice set. It is not necessary because in reality the full size is almost infinite. This result also shows that when the sampling size is far less than the full size of choice set, for example, 2 VS.20, the number of generated paths is close to the sampling size. It proves that the capacity of choice set is much important for the choice set generation process and then heavily affects the latent modeling process. We can also conclude that in a real network the final result, the capacity of choice set, depends on the sampling size.

- Analysis 2: What is the relationship between the size of choice set and the original value paths?

  Full size of path: 20  
  Experiment times: 1000  
  Sampling size: 20  
  Cost of link: x for 19 links and 0.9 for other 5 links metioned above

  In this analysis, we want to test how the original differences among paths will affect the final generation result. We set the cost of 5 links in a typical path as a constant value and set the cost of other links various. We expect that when the cost of the 19 links is much larger than 0.9, the generation process will treat this typical path as the minimum cost path and thus the capacity of paths will be close to a very small number.

  The result shows that this expectation is true. It means that if a path has a very high cost than other paths, it will have low probability to be chosen by the random weight generation. This is not only true for the path but also for the link.
Figure 3-18: The analysis of size of choice set and the original value of path
Chapter 4

Case Study: Boston

In this chapter, I will use the methodology mentioned above to do a research on Boston. The whole framework is the same as traditional route choice analysis, which contains data processing, choice set generation and choice modeling.

In section 4.1, I will describes the sources of data used in the whole process. It includes the introduction of data range and attributes. In section 4.2, I will describe the data processing from the initial data which is the GPS points and network data in GIS format, to the polished data that can be directly used in the choice set generation model. In section 4.3, I will use the method developed in Chapter 3 to generate the alternative paths. In section 4.4, I will develop the route choice model using Path Logit Size model.

4.1 Data Description

There are two types of data in this research, the GPS data of car and the network data of Boston area.

The initial GPS data is the car data. In this thesis, the useful information of it are: GPS time, Car ID, latitude and longitude (see table 4.1). Though this data also contains the speed for the observation point, it is very volatile. Many value are not realistic, it is not used in this research. The example of this initial data is shown in the table 4.2.
Field | Description and example
--- | ---
GPS time | the time of the observation using UNIX time format, for example, 1252765825;
Car ID | the id of the car, for example, 00:18:0a:01:68:03;
Latitude | the latitude of observation point, for example, 42.400048
Longitude | the longitude of observation point, for example, -71.062406.

Table 4.1: GPS Data

<table>
<thead>
<tr>
<th>Car ID</th>
<th>GPS Time</th>
<th>Latitude</th>
<th>Longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:18:0a:01:68:03</td>
<td>1252765826</td>
<td>42.400048</td>
<td>-71.062406</td>
</tr>
<tr>
<td>00:18:0a:01:68:03</td>
<td>1252765827</td>
<td>42.400049</td>
<td>-71.062405</td>
</tr>
<tr>
<td>00:18:0a:01:c4:e9</td>
<td>1252839012</td>
<td>42.601411</td>
<td>-70.967518</td>
</tr>
<tr>
<td>00:18:0a:01:c4:e9</td>
<td>1252839087</td>
<td>42.58695</td>
<td>-70.966119</td>
</tr>
<tr>
<td>00:18:0a:01:c4:e9</td>
<td>1252839088</td>
<td>42.586786</td>
<td>-70.966189</td>
</tr>
</tbody>
</table>

Table 4.2: Examples of raw data

The raw data has some outliers that should be eliminated. The typical error of it is the location error, some of which are even located in Europe. This should be the systematic error of GPS device. It is shown in these figures (4-1) and (4-2).

After removing outliers according to the location of observation points, the data looks much better. We also match it into the Google Earth to exam the whole data. In Google Earth these points match the road network accurately. The range of latitude and longitude is shown as figure4-3 and figure4-4: The road network data is downloaded from the website of GIS laboratory in MIT. The original data contains
the whole Massachusetts data, not just Boston. In this research, we want to limit the research area around downtown Boston and part of Cambridge near the Charles River for a further research in combination of cell phone data, and also for an observation of travel behavior of automotive vehicle in downtown area. As we mentioned above in Chapter 3, the travel behavior between long distance and short distance is different. We cannot observe the overall travel behavior by just considering a small part of area, but we can observe the behavior for a specific circumstance. The network data includes not only the physical attributes of network such as the distance of link, but also the topological attributes, which represents the relationship among all links. Besides, it also includes the road type information. There are 17 kinds of road types, including trunk road, primary road, secondary road, pedestrian and so on. Since the GPS data is from the car, some road type are not allowed to used in the research,
such as the pedestrian and cycle path.

4.2 Data Processing

In this part, I will process the raw data and let it be the data that can be directly used in the model specification. The model specification requires a table which contains the revealed paths and alternative paths and their attributes such as distance and road type. However, the network data in GIS format is difficult to use with the combination of GPS data. One reason is that the network data in GIS cannot be used directly in the choice set generation process. Another reason is that the GPS data is a sequence of points which needs to be matched using network data and to extract the attribute information from it. The general solution for it is to transfer all these two types of data into the data structure of graph. In this situation, the choice set generation algorithm is easily to be implemented and the attribute information is also easy to extract.

4.2.1 Transforming Network Data

The original GIS data is the ESRI Shapefile data format described in polyline geometry. The general type and example of this data is shown in the table 4.3. The object data structure is mentioned in the equation 3.5 and 3.4 in chapter 3. To form a graph data structure, we need to generate two end points of each link. Then for any links which share the same end point, keep one point and delete others. The judgment of

<table>
<thead>
<tr>
<th>Line ID</th>
<th>Road Type</th>
<th>Road Name</th>
<th>Oneway</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>cycleway</td>
<td>Dr. Paul Dudley White Path</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>secondary</td>
<td>Beacon Street</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>teriary</td>
<td>Magazine Street</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>primary</td>
<td>Commonwealth Avenue</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>primary</td>
<td>Memorial Drive</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.3: Initial Network Data
identical points is on the basis of their latitude and longitude. If their locations are the same, they are treated as identical points and then keep one of them. The result shows as the figure 4-6.

4.2.2 Transforming GPS Data

The processing of GPS data is much more complicated than the network data. The whole process includes trajectory segmentation and map matching work. The original GPS data are several car trajectories which record the whole movement in around 15 days. We need to extract a number of paths from a trajectory because the trajectory lasts a long period which includes all movements. An example is shown in the figure 4-8. The x-axis and y-axis represents its longitude and latitude. Downtown Boston is in the dense part of the trajectory. In figure 4-9 we can observe the change of the latitude, longitude and the speed. A direct sense may think of dividing the trajectory according to the pattern of speed, which seems to be at a relative constant level in a trip. In this research, we found that the method is not appropriate for the reason of a heavy volatility.

However, we find a better and simpler way to implement it. The time gap between two consecutive points shows a pattern for helping the trajectory segmentation. This may due to the use of GPS device in the taxi business. We set a threshold $t$ and if the time gap between two consecutive points is larger than $t$, the ahead point should
be the end point of a trip and the latter point should be the start point of another trip. In this thesis, we run a sensitivity analysis to observe how the number of paths will be affected by the time gap. The result (figure 4-7) shows that the number of paths (y-axis) drops dramatically with the increasing of time gap when the time gap is small. On the contrary, it has less sensitive when the time gap is large. This should partly due to the long period observation which contains a number of overnight time. Another important reason is the irregular record in small time gap that should be the characteristics of GPS device. Hence, we need to find an appropriate and feasible time gap for separating trajectories. In this research, we define 5-minute as the time threshold.
4.2.3 Map Matching

After the trajectory segmentation, the following step is the map matching. It deals with the GPS and GIS data at the same time. The purpose of it is to transform the format of GPS data into the data format of network data so that the trace of individuals can be presented in the road network rather than a sequence of geographical points. The method of map matching is mentioned in Chapter 3. In general, these GPS points will be matched to the closest node in the network by changing their latitude and longitude value (as Equation 4.1).

\[(Point_1, Point_2...Point_i) \rightarrow (Node_1, Node_2..., Node_j)\]  \hspace{1cm} (4.1)

Notice, here \(i\) should smaller than \(j\) because some of consecutive points will be matched to the same node. In this case, these points are identical and they should be kept only one points and delete others.

Since there are about 300,000 GPS points and about 10,000 network nodes in this Boston case study, the computational time is an inescapable issue. To solve this problem, I use the Local Searching Method to improve the efficiency. This model will divide the GPS points and the nodes of network into many small regions separately in the first step. Then it will search the closest node for each GPS points. In the second step, the model will search the closest node for each GPS from the neighborhood region of the GPS region. If the distance of closest node in the neighborhood region...
is smaller than in the same region, it will be accepted. In this case study, the whole region is divided into 110 \((10 \times 11)\) regions. This greatly reduces the computational time to about 1/10 of pre-optimization result. Besides, the quality of GPS data is enough to allow us to ignore the second step of Local Searching Method. In this case, the computational time is less than 1/100 of pre-optimization result with a very small tradeoff of data quality of final result.

An example of map matching for a specific path with loop is shown in the figure4-12 and figure4-13. In this example, the revealed path is composed of a lot of GPS points(figure4-12). Most of these points are close to the nodes of road network but not the same as them. After the map matching work, the number of GPS points is reduced to a small number while at the same time their latitude and longitude value are changed to those of nodes in the network. In a word, these GPS points are replaced with network nodes.

### 4.3 Route Choice Set Generation

Until now, we have the revealed path for each OD pair. In this route choice set generation process, we will extract the OD pair information from the revealed paths and use the Random Weight Path Generation Model to create alternative paths for them. In fact, alternative paths are virtual and are affected by the socioeconomic attribute of each individual. Since the kind of data in the Boston case study is not
available, I will just simply generate the alternative paths by just using the network data.

As mentioned in Chapter 3, the Random Weight Choice Set Generation Model is a shortest-path based model with the combination of random weight for the attribute of links. For each link $i$ of individual $n$ we have the cost function as the equation 3.12. In this case we do not consider the differences of cost function among individuals because we have no availability on this data. This cost function is:

$$C_i = w_1X_{i1} + w_2X_{i2} + ... + w_kX_{ik}$$ (4.2)

These meaning of notations are the same as equation 3.12. In this research, since we only have the distance and road type information as the attribute of links, we have to use the distance as the only variable for the cost function. It is also appropriate because the travel time in a dynamic variable which is closely related to distance of links. So we set the cost function of links as:

$$C_i = wX_{dist}$$ (4.3)

Also, we need to decide the distribution of random weight of each link. Due to the data limitation, we can just assume that the mean value, variance and the distribution form of random weight of each link is identical. It doesn't mean their values are the same since they are random numbers. In this case, I set it as the normal distribution with 0 as the mean value and 1 as the variance:

$$w = e^{N(0,1)}$$ (4.4)

After generating random numbers of each link, we have a new network with different attributes compared with the original network. Under the circumstance of new network, we can use the shortest path algorithm to generate the minimum-cost path for each OD pair. This process is repeated for $n$ times and we get $n$ paths for a specific OD pair. Since these $n$ shortest paths are generated in different situation,
some of them may be identical. Keep just one from these identical paths. The figure 4-14, 4-15, 4-16, 4-17 and 4-18 are the examples of an OD pair from Boston University Bridge to the Longfellow Bridge. Figure 4-14 is the revealed path extracted from GPS data. It travels through Vassar Street, Massachusetts Avenue, and then crosses Harvard Bridge and travels along the Storrow Drive to the destination. The figure 4-15 is an alternative path that travels through Vassar Street, Massachusetts Avenue, then turn left to the Memorial Drive. The figure 4-16 travels along the Vassar Street and turns to the Main Street. The figure 4-17 is similar to the figure 4-16 but it uses the Portland Street as the connection between Vassar Street and Main Street. The last figure 4-17 doesn’t use the Vassar Street as its main path but some small paths. These paths are representative for the real travel.

Some violations may happen such as the one way and left turn restriction. Though there is a variable called “oneway” in the origin GIS data, we don’t know its direction yet. The left turn information is absent in the network data too. If the network data contains more information, the Choice Set Generation model can also add these kinds of restrictions into the model.

There are also some problems in the overpass area around downtown in the network data. In this area there are many highways in different height. Some of them are not connected even when they are the same location. But the GIS data has no information about the height and treats them connected in the same plane. This may cause some bias in the path generation process because there are some intersections in the graph and they don’t exist actually. If data supports, it is very easy to build a graph since we only need to construct links for those roads which are in connection in the real world.
Figure 4-14: An Example of Revealed Path

Figure 4-15: The First Example of Alternative Path

Figure 4-16: The Second Example of Alternative Path

Figure 4-17: The Third Example of Alternative Path

Figure 4-18: The Fourth Example of Alternative Path
In here, we can also check the information of distance and links of the alternative path. The figure 4-19 shows the histogram of alternative paths for the specific OD pair discussed above. The figure 4-20 shows the histogram of number of links in path. They prove that the algorithm works well for the coverage and the representativeness. If the algorithm creates paths with too much similarity, these numbers will distribute in a narrow range.

After these steps, the remained work is to generate variables for revealed paths and alternative paths. These variables include: distance, link number, path size value, the proportion of primary road, the proportion of trunk road, the proportion of secondary road and the proportion of motorway.

The general data flow of these steps is shown in the figure 4-21.
Figure 4-21: Data Flow of the Route Choice Analysis
4.4 Model Estimation

In this part, I will use the PSL model to estimate the parameters of variables. Plus, I will also use the MNL model to repeat it and then compare these two models.

The PSL model and the MNL model are specified with the same parameters of the deterministic part of the utility function. The deterministic part $V_i$ of the utility for path $i$ is

$$V_i = \beta_{ps} \ln (PS_i) + \beta_{distance} \text{PathDistance}_i + \beta_{link} \text{LinkNumber}_i + \beta_{primary} \text{PrimaryRoad}_i + \beta_{secondary} \text{SecondaryRoad}_i \quad (4.5)$$

Here the PS attribute is only contained in the PSL model. The MNL model does not have this part. In the deterministic part I introduce the distance of path ($\text{PathDistance}_i$), the number of links in path ($\text{LinkNumber}_i$) the proportion of primary road of the path ($\text{PrimaryRoad}_i$) and the proportion of the secondary path ($\text{SecondaryRoad}_i$) as the variables in the model. The distance of a path is computed by adding up all length of links in the path. The number of links is computed by counting the number of link in the path. The proportion of primary road and secondary road are computed separately by adding up the proportion of length of these road types of the path. Here the PS value is calculated based on the length of links for the reason of data limitation and the convenience of calculation. The formulation 2.13 is introduced in Chapter 2. Before model estimation, I also deal with heteroscedasticity by specifying different scale parameters for different cars. Statistics on these attributes in the model specification are shown in Table 4.4.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Min</th>
<th>Max</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>LinkNumber</td>
<td>6</td>
<td>480</td>
<td>84.88</td>
</tr>
<tr>
<td>Distance(km)</td>
<td>0.332</td>
<td>16.9064</td>
<td>3.143</td>
</tr>
<tr>
<td>PrimaryRoad</td>
<td>0</td>
<td>0.9709</td>
<td>0.1874</td>
</tr>
<tr>
<td>SecondaryRoad</td>
<td>0</td>
<td>1</td>
<td>0.3702</td>
</tr>
<tr>
<td>Ln(PS)</td>
<td>-15.216</td>
<td>0</td>
<td>-2.644</td>
</tr>
</tbody>
</table>

Table 4.4: Statistics on Attributes
The final number of observation route is 244 in this case. It is enough to estimate the model. Here I use BIOGEME\(^1\) for all model estimations. The estimation result in table 4.5 is quite reasonable.

Firstly, the parameter of distance shows no significantly different from zero in both two models. In MNL, it is positive while in PSL it is negative. This is because the car shows little preference for the short distance in a congested urban network. The short distance in the congested environment is not so attractive because most of short-distance routes are composed of many branch ways. Another reason is that in a small urban area, such as this case, the travel time is affected by other factors, especially the average speed of road and the traffic condition.

The link number affection is beyond expectation because I expect that the less number of links in a path will be more attractive. One reason to explain this is that the link of original GIS data is not built on the basis of roads between two intersections. Instead, the original GIS data contains more links between two road intersections. On the contrary, the link in a typical transportation graph should represent the actual road between two road intersections. Another reason maybe that drivers tend to travel around everywhere to run their business so that their visit will cover more roads. This results in a preference of high link number in the route choice. This is very logical because I also notice that the secondary road is the most attractive road type for the car.

Before choosing the proportion of primary road and the secondary road, I also try other road types. It shows that only the proportion primary road and the secondary road significantly affect the route choice, especially for the secondary road. The secondary road is a major road type of urban area. In the transportation network, it plays as the role of the blood capillary in human body. The secondary road may also relate to the running business of these cars so that drivers prefer to choose a secondary road for picking up passengers rather than running on the trunk road with no business. Besides, the result shows that drivers would like to use the primary road. In this Boston case, the Memorial Drive is the major part of this road type. Its

\(^1\)http://transp-or.epfl.ch/page63023.html
connectivity is extremely important for Boston since there are just several bridges, which are linked with Memorial Drive, connecting the south side and north side of Charles River.

<table>
<thead>
<tr>
<th></th>
<th>MNL</th>
<th>PSL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LinkNumber(β)</td>
<td>0.097</td>
<td>0.0845</td>
</tr>
<tr>
<td>LinkNumber(t-stat)</td>
<td>12.94</td>
<td>9.4</td>
</tr>
<tr>
<td>Distance(β)</td>
<td>0.313</td>
<td>-0.0213</td>
</tr>
<tr>
<td>Distance(t-stat)</td>
<td>1.49</td>
<td>-0.09</td>
</tr>
<tr>
<td>Primary(β)</td>
<td>3.08</td>
<td>2.80</td>
</tr>
<tr>
<td>Primary(t-stat)</td>
<td>3.47</td>
<td>2.74</td>
</tr>
<tr>
<td>Secondary(β)</td>
<td>2.85</td>
<td>4.02</td>
</tr>
<tr>
<td>Secondary(t-stat)</td>
<td>3.40</td>
<td>4.28</td>
</tr>
<tr>
<td>Ln(PS)(β)</td>
<td>-</td>
<td>2.76</td>
</tr>
<tr>
<td>Ln(PS)(t-stat)</td>
<td>-</td>
<td>15.09</td>
</tr>
</tbody>
</table>

Table 4.5: Estimation Results

If we compare the result of MNL model and the PSL model, the PSL model has more advantages in the route choice analysis. From the table 4.6 we know that the prediction power of PSL model is much better than the MNL model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Nb.estimate parameters</th>
<th>Final log-likelihood</th>
<th>adjusted rho-square</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNL</td>
<td>4</td>
<td>-536.610</td>
<td>0.434</td>
</tr>
<tr>
<td>PSL</td>
<td>5</td>
<td>-265.275</td>
<td>0.717</td>
</tr>
</tbody>
</table>

Table 4.6: Model Fit Measures

After constructing the model, we can use it to predict individual behaviors. The model is the core part of the whole prediction and simulation. From the model, we can use it to test the probability of individuals choosing a specific route. Also, we can use it to test the importance of a specific road. For example, we remove the link representing a sudden close of Harvard Bridge. Under the situation of identical travel demand, how will the traffic flow change in a network with and without the Harvard Bridge? Overall, the performance of predicting choice probability of the model is a key role in implementing all kinds of these tasks.
Chapter 5

Conclusion

5.1 Summary

This thesis discusses about the usage of GPS data in route choice analysis.

The GPS data processing is a bright direction but also a tough step for the transportation research. In this thesis, I show a detailed illustration on how to process it from the raw data to the final data which can be implemented directly in the model estimation. The trajectory segmentation is the first step in the process. I use the time interval as the "scissors" to cut a completed trajectory into several paths. After that I match the GPS data to the network data which is obtained from GIS data because it has very good quality on the road network. To solve the computational issue, I develop the point-based local searching map matching way to improve the map matching efficiency. This method includes two steps. The first step is to search the in-region points for each GPS point. The second step is to search the neighborhood region to avoid some possible bias.

The random weight method is then developed in the choice set generation. This is a method on the basis of shortest path algorithm as other choice set generation models. However, this method has a basic idea that the utility for individuals is always changing and should be considered. To find alternative paths for a revealed path, the model generate random numbers for the network and then use the shortest path algorithm to find the "minimum cost" path which is calculated with the random
weight. In this research, I just use the distance attribute due to the data limitation. I generate random weights for it and get a new value of distance. The “minimum cost” path is found on the basis of new values. After repeating this work in many times for different OD pairs, the whole choice set is completed. Then the attribute of these paths is calculated on the basis of original value rather than the new value. It shows a very good coverage in the road network and has an advantage on the computational efficiency.

In the model specification and estimation part, I use the Path Size Logit model to estimate the data. I specify some attributes, such as distance, link number and the proportion of trunk road in the whole path, as the variables that affect the choice of paths. The result is quite reasonable and the explanation power of model is enough. Also, I use the MNL model to repeat the process with the same variables. The comparison shows that Path Logit Size model is an improvement of MNL model.

In general, this thesis provides a comprehensive research on the route choice analysis from data processing to the model estimation. It also includes some examples to show the intermediate result for better understanding.

5.2 Future Research

Future directions are suggested:
• A further research applied with the random weight method in the modeling process. In this thesis, the random weight method is only used in the path generation process. However, the basic idea of random weight is feasible for the modeling because the urban transportation system is a dynamic and time-dependent system. For example, the travel time for a trunk road can vary in rush hour and normal hour. People may expect this situation and change their behavior to adapt it. In this case, the stochastic characteristics is imbedded in the utility part, opposed to the traditional model. As we know, the traditional models use random part to capture the unknown variable or some errors, but the utility part is still deterministic. Thus, this direction maybe very interesting and useful since it is much more related to the real world. In a further
research, the utility part in model can be separated into two parts, the deterministic part and the stochastic part. An example in route choice is that the distance and road type is deterministic while the travel time is stochastic in the utility function.

- A further research on the relationship of among random weights. In this thesis, I just set an identical random weight for each link. However, this is a simplified assumption due to the limitation of data. An example is that a trunk road is connected with a secondary road. If the trunk road has a heavy traffic flow, it will cause some flows to the secondary road. Furthermore, there should be some patterns in the relationship among these two roads. Maybe the trunk road has less impact on the secondary road with the increasing of traffic flow in trunk road when the trunk road has a relative small flow. Thus, some probability models can be introduced when setting the random weight for each link. This research will be very useful in the traffic assignment and the traffic simulation.

- Estimation result and efficiency comparison in different models. Up to today, there are a bunch of models developed in the route choice analysis. A comparison of these models should be necessary since they are designed to solve different problems and are used in different scenarios. An comprehensive comparison is useful for us. It tells us which model has the best estimation result, which model has the best computational efficiency that is useful in large-scale network, which model is appropriate for multi-modal travels, and so on. Also, their drawbacks and limitations should be listed.

- A deep research on the relationship between the topology of road network and the route choice behavior. In Chapter 3, I point out that the number of alternative choices is affected by the topology type of network. For the simplicity, I set a grid network in the research. An improvement of this assumption can help better give an appropriate number of alternative paths. In theory, some topology knowledge can be applied to solve it.

- A further usage in a variety of data. Nowadays, a lot of travel information is available in the website, especially for the up-to-date location-based social networking website. These data are very attractive for most of them are highly connected with
other behaviors and the socioeconomic attributes. This information is rare in GPS form since GPS has no way in telling us about these except the travel information. If these data are available for the route choice analysis, it can help us better understand and predict the travel behavior.

- The further simulation work on the traffic flow. After estimating parameters of the model, some practical research can be implemented on the basis of the model, especially the simulation process.
Appendix A

Matlab Code

- Path Segmentation

clear all, close all, clc

%% load data and set them into different variables
fid=fopen ('gpsdata.txt');
data=textscan(fid,'XfsYffff','delimiter',',','Headerlines',0);
close(fid);
time=data{1};
id=data{2};
lat=data{3};
long=data{4};
stop=data{5};
speed=data{6};

%% delete the out-of-area data
outlier_data = find(lat > 42.3641);
time(outlier_data) = [];
id(outlier_data) = [];
lat(outlier_data) = [];

77
long(outlier_data) = [];
stop(outlier_data) = [];
speed(outlier_data) = [];
outlier_data = find(lat < 42.3425);
time(outlier_data) = [];
id(outlier_data) = [];
lat(outlier_data) = [];
long(outlier_data) = [];
stop(outlier_data) = [];
speed(outlier_data) = [];
outlier_data = find(long > -71.0495);
time(outlier_data) = [];
id(outlier_data) = [];
lat(outlier_data) = [];
long(outlier_data) = [];
stop(outlier_data) = [];
speed(outlier_data) = [];
outlier_data = [];
outlier_data = find(long < -71.1103);
time(outlier_data) = [];
id(outlier_data) = [];
lat(outlier_data) = [];
long(outlier_data) = [];
stop(outlier_data) = [];
speed(outlier_data) = [];

%% divide the data according to taxi
id_unique = unique(id);  %% list of unique users
for taxi_id = 1:length(id_unique)
    index_gps = [];

78
index_gps = find(strcmp(id, id_unique(taxi_id))); 
eval(['time_taxi' num2str(taxi_id) ' = time(index_gps)']); 
eval(['lat_taxi' num2str(taxi_id) ' = lat(index_gps)']); 
eval(['long_taxi' num2str(taxi_id) ' = long(index_gps)']); 
eval(['stop_taxi' num2str(taxi_id) ' = stop(index_gps)']); 
eval(['speed_taxi' num2str(taxi_id) ' = speed(index_gps)']);
end

%%for taxi 1

time_sorted = [];
time_sorted_index = [];
[ttime_sorted time_sorted_index] = sort(time_taxi1);
for no_of_row = 1:length(time_sorted_index)
    if no_of_row < length(time_sorted_index)
        time_gap1(time_sorted_index(no_of_row),1) = time_taxi1(time_sorted_index(no_of_row),1)
    else
        time_gap1(time_sorted_index(no_of_row),1) = 0;
    end
end

count_point = 1;
count_path = 1;
for no_of_row = 1:length(time_sorted_index)
    if time_gap1(time_sorted_index(no_of_row),1) <= 300
        path1(time_sorted_index(no_of_row),1) = count_path;
        point1(time_sorted_index(no_of_row),1) = count_point;
        count_point = count_point + 1;
    else
        path1(time_sorted_index(no_of_row),1) = count_path;
        point1(time_sorted_index(no_of_row),1) = count_point;
        count_path = count_path + 1;
end
count_point = 1;
end
end

%% delete the fraction of path
%% taxi 1
for path_number = 1:length(find(unique(path1)))
    index_path = find(path1 == path_number);
    if length(index_path) <= 200
        time_taxi1(index_path) = [];
        time_gap1(index_path) = [];
        lat_taxi1(index_path) = [];
        long_taxi1(index_path) = [];
        stop_taxi1(index_path) = [];
        speed_taxi1(index_path) = [];
        path1(index_path) = [];
        point1(index_path) = [];
    else
        index_path = [];
    end
end

length(unique(path1))+length(unique(path2))+length(unique(path3))+length(unique(path4))
%% delete the unnecessary points using keep_sequence matrix
final_lat_taxi1 = lat_taxi1(keep_sequence1);
final_long_taxi1 = long_taxi1(keep_sequence1);
final_path1 = path1(keep_sequence1);
final_point1 = point1(keep_sequence1);
final_speed_taxi1 = speed_taxi1(keep_sequence1);
final_stop_taxi1 = stop_taxi1(keep_sequence1);
%% plot the path 1 for taxi 1
index_path = find(path1 == 3);
[point_sorted index_point_sorted] = sort(time_taxi1(index_path));

figure
plot(long_taxi1(index_path(index_point_sorted)),lat_taxi1(index_path(index_point_sorted)));
xlabel('long'),ylabel('lat')

figure
subplot(311)
plot(time_taxi1(index_path(index_point_sorted)),long_taxi1(index_path(index_point_sorted)));
subplot(312)
plot(time_taxi1(index_path(index_point_sorted)),lat_taxi1(index_path(index_point_sorted)));
subplot(313)
plot(time_taxi1(index_path(index_point_sorted)),speed_taxi1(index_path(index_point_sorted)));

kmlwrite(['Path' num2str(4)],lat_taxi1(index_path),long_taxi1(index_path))

%%% generate KML files for taxi 1
rest_path = unique(path1);
for path_number =1:length(rest_path)
    index_path = find(path1 == rest_path(path_number));
    [point_sorted index_point_sorted] = sort(time_taxi1(index_path));
    kmlwrite(['Path1_' num2str(rest_path(path_number))],lat_taxi1(index_path),long_taxi1(index_path))
end

%%% sensitivity analysis of number of path affected by time interval
no_of_gap=[];
for find_right_gap = 60:3600
    no_of_gap(find_right_gap-59,1) = length(find(time_gap1 > find_right_gap)) + len
end
• Path Size Calculation

```
%%taxi1
%%for revealed path
temp_path = unique(OD_table_final_taxi1(:,15));%%read the index of path
for i=1:length(temp_path)
    index_path_point = find(final_compact_path_point(:,1) == temp_path(i));
    temp_link = unique(final_compact_path_point1(index_path_point,7));%%read the in
    temp_all_length = sum(final_compact_path_point1(index_path_point,8));
    sum_ps = 0;%%
    for j = 1:length(temp_link)
        if temp_link(j)==0
            temp_link_point = find(final_compact_path_point1(:,1)==temp_path(i) & f
            temp_sum = sum(final_compact_path_point1(temp_link_point,8))/temp_all_l
            sum_ps = sum_ps + temp_sum;
        else if temp_link(j)==99999;
            temp_sum = 0;
            sum_ps = sum_ps + temp_sum;
        else
            temp_path_point_in_alternative = find(algo_final_path_point1(:,1)==
            temp_no_of_path_shared = length(find(algo_final_path_point1(temp_pa
            temp_no_of_path_in_revealed = find(final_compact_path_point1(index_
            temp_sum = final_compact_path_point1(temp_no_of_path_in_revealed(1)
            sum_ps = sum_ps + temp_sum;
        end
    end
end
OD_table_final_taxi1(i,16) = log(sum_ps);
end
```

%%taxi1
%%for alternative path

\[
\text{temp\_route} = \text{unique}(\text{OD\_final\_algo\_taxi1}(:,13));
\]

for \( i = 1 : \text{length} (\text{temp\_route}) \%

\[
\text{index\_temp\_route} = \text{find}(\text{OD\_final\_algo\_taxi1}(:,13) == \text{temp\_route}(i));
\]

\[
\text{temp\_path} = \text{unique}(\text{OD\_final\_algo\_taxi1}(\text{index\_temp\_route},14));
\]

for \( j = 1 : \text{length} (\text{temp\_path}) \%

\[
\text{index\_temp\_path} = \text{find}(\text{OD\_final\_algo\_taxi1}(\text{index\_temp\_route},14) == \text{temp\_path}(\text{index\_temp\_path}));
\]

\[
\text{temp\_path\_point} = \text{find}(\text{algo\_final\_path\_point1}(1,:) == \text{temp\_route}(i) \& \text{algo\_final\_path\_point1}(\text{temp\_path\_point},1) == \text{temp\_path}(\text{index\_temp\_path}));
\]

\[
\text{temp\_link} = \text{unique}(\text{algo\_final\_path\_point1}(\text{temp\_path\_point},1) == \text{temp\_route}(i) \& \text{algo\_final\_path\_point1}(\text{temp\_path\_point},7) == \text{temp\_link});
\]

\[
\text{temp\_all\_length} = \text{sum}(\text{algo\_final\_path\_point1}(\text{temp\_path\_point},1) == \text{temp\_route}(i) \& \text{algo\_final\_path\_point1}(\text{temp\_path\_point},9) == \text{temp\_link});
\]

\[
\text{sum\_ps} = 0;
\]

for \( k = 1 : \text{length} (\text{temp\_link}) \%

\[
\text{if} \ \text{temp\_link}(k) == 99999;
\]

\[
\text{temp\_sum} = 0;
\]

\[
\text{sum\_ps} = \text{sum\_ps} + \text{temp\_sum};
\]

else

\[
\text{temp\_no\_of\_path\_shared\_in\_alternative} = \text{length}(\text{find}(\text{algo\_final\_path\_point1}(1,:) == \text{temp\_route}(i) \& \text{algo\_final\_path\_point1}(\text{temp\_path\_point},1) == \text{temp\_link}));
\]

\[
\text{temp\_no\_of\_path\_shared\_in\_revealed} = \text{length}(\text{find}(\text{final\_compact\_path}(1,:) == \text{temp\_route}(i) \& \text{final\_compact\_path}(\text{temp\_path\_point},1) == \text{temp\_link}));
\]

if \( \text{temp\_no\_of\_path\_shared\_in\_revealed} == 0 \)

\[
\text{temp\_no\_of\_path\_shared\_in\_revealed} = 0;
\]

else

\[
\text{temp\_no\_of\_path\_shared\_in\_revealed} = 1;
\]

end

\[
\text{temp\_no\_of\_path\_shared\_in\_all} = \text{temp\_no\_of\_path\_shared\_in\_alternative} + \text{temp\_no\_of\_path\_shared\_in\_revealed};
\]

\[
\text{index\_temp\_link} = \text{find}(\text{algo\_final\_path\_point1}(1,:) == \text{temp\_link}(1));
\]

\[
\text{temp\_sum} = \text{algo\_final\_path\_point1}(\text{index\_temp\_link}(1),9)/\text{temp\_all\_length} - \text{sum\_ps} = \text{sum\_ps} + \text{temp\_sum};
\]

end
temp_index_in_OD = find(OD_final_algo_taxi1(:,13)==i & OD_final_algo_taxi1(OD_final_algo_taxi1(temp_index_in_OD,15) = log(sum_ps);
end
end

- Map Matching

%%%Match the path to the network.
%%%final_lat/long-->matched_lat/long-->compact_path_point

%%%define the region for map points
for i = 1:length(node)
a = fix((node(i,3)+71.1103)/(-71.0495+71.1103)*10)+1;%%%transform long
b = fix((node(i,4)-42.3425)/(42.3641-42.3425)*10);%%%transform lat
node(i,6) = a+b*10;%%%long and lat
end

%%%define the region for GPS points
for i = 1:length(finaljlattaxil)
a = fix((final_long_taxi1(i,1)+71.1103)/(-71.0495+71.1103)*10)+1;
b = fix((final_lattaxi1(i,1)-42.3425)/(42.3641-42.3425)*10);
point_loc_taxi1(i,1) = a+b*10;
end

%%%find the closest point for GPS points from map points
for i = 1:length(point_loc_taxi1)
temp_row = [];
temp_distance = [];
temp_row = find(node(:,6) == point_loc_taxi1(i,1));
for j = 1:length(temp_row)
    [temp_distance(j,1),temp_distance(j,2)] = distance(final_lat_taxi1(i,1),final_l end

row_of_min_distance = [];
min_dist = min(temp_distance(:,1));
row_of_min_distance = find(temp_distance(:,1) == min_dist);

match_taxi1(i,1) = node(temp_row(row_of_min_distance(1)),4);
match_taxi1(i,2) = node(temp_row(row_of_min_distance(1)),3);
end

%% matching for taxi1
find_path = unique(final_path1);
index_of_path_in_taxi = 0;%%the index of paths
number_of_lines_in_taxi = 0;%% calculate the overall lines of table for a taxi
for i = 1:length(find_path)%%i is the index of path
    path_index = [];
    path_index = find(final_path1 == find_path(i));
    [point_sorted point_sorted_index] = sort(final_point1(path_index));

    index_of_points_in_path = 0;%%the index of points

    for j = 1:length(path_index)%%j is the index of point for path i
        if j == 1
            index_of_path_in_taxi = index_of_path_in_taxi + 1;
            index_of_points_in_path = index_of_points_in_path + 1;
            number_of_lines_in_taxi = number_of_lines_in_taxi + 1;
            compact_path_point1(number_of_lines_in_taxi,1) = index_of_path_in_taxi;

85
compact_path_point1(number_of_lines_in_taxi,2) = index_of_points_in_path
compact_path_point1(number_of_lines_in_taxi,3) = match_taxi1(path_index
compact_path_point1(number_of_lines_in_taxi,4) = match_taxi1(path_index
else
    if match_taxi1(path_index(point_sorted_index(j)),1) == compact_path_point
        number_of_lines_in_taxi = number_of_lines_in_taxi + 0;\%do nothing
else
    number_of_lines_in_taxi = number_of_lines_in_taxi + 1;
    index_of_points_in_path = index_of_points_in_path + 1;
    compact_path_point1(number_of_lines_in_taxi,1) = index_of_path_in_t
    compact_path_point1(number_of_lines_in_taxi,2) = index_of_points_in
    compact_path_point1(number_of_lines_in_taxi,3) = match_taxi1(path_i
    compact_path_point1(number_of_lines_in_taxi,4) = match_taxi1(path_i
end
end
end
end

\%exam a specific matching in google earth
test_yuanlai = find(final_path1==9);
kmlwrite(['Test_yuanlai'],final_lat_taxi1(test_yuanl)
test_houlai = find(compact_path_point1(:,1)==5);
kmlwrite(['Test_houlai'],compact_path_point1(test_houlai,3),compact_path_point1(tes

\%find the node ID for these table
for i = 1:length(compact_path_point1)
    compact_path_point1(i,5) = find(compact_path_point1(i,3)==node(:,4) & compact_p
    temp_index_point = find(graph(:,3) == compact_path_point1(i,5) | graph(:,4) ==
    if length(temp_index_point) > 0
        compact_path_point1(i,6) = graph(temp_index_point(1),2);\%road type

86
else
    compact_path_point1(i,6) = 999;\% the value of unknown road type is 999
end
end

temp_path = unique(compact_path_point1(:,1));
for i = 1:length(temp_path)
    temp_point = find(compact_path_point1(:,1) == i);\% find points of a path
    for j = 1:length(temp_point)
        if j == length(temp_point)\% the last point
            compact_path_point1(temp_point(j),7) = 99999;\% the last point
        else
            if length(find((compact_path_point1(temp_point(j),5) == graph(:,3) & co
                compact_path_point1(temp_point(j),7) = 0;\% not in the existing graph
            else
                temp_link = find((compact_path_point1(temp_point(j),5) == graph(:,3)
                compact_path_point1(temp_point(j),7) = temp_link(1);
            end
        end
    end
end
end

\% compute the link distance
\% taxi1

temp_path = unique(compact_path_point1(:,1));
for i = 1:length(temp_path)
    temp_point = find(compact_path_point1(:,1) == i);\% find points of a path
    for j = 1:length(temp_point)
        if j == length(temp_point)
compact_path_point1(temp_point(j),8) = 0;

else

    [temp_dist,temp_az]=distance(compact_path_point1(temp_point(j),3),compa
temp_distance = temp_dist*1.1078e+005;
    compact_path_point1(temp_point(j),8) = temp_distance;

end
end
end

• Path Generation

• Path Generation

  % algo_path_point is the table generated by the model.
  % routeID, pointID,lat,long,nodeID,roadtype,linkID,pathID

  % taxi1
  row_in_OD_algo = 0; % initialize; the row index for each alternative path
temp_path = 0; % initialize; the index of path
temp_point = 0; % initialize; the index of point in path
algo_path_point1 = [];
for i = 1:length(OD_table_taxi1)
    temp_path = temp_path+1;

    for j = 1:100 % repeat times

        shortest_dist = [];
        shortest_path_transpose = [];
        row_in_OD_algo = row_in_OD_algo + 1;


88
rand_distance(1:length(graph),1) = exp(randn(length(graph),1)) .* graph(1:length(graph),1)
G_rand = sparse(graph(:,3),graph(:,4),rand_distance(:,1),length(node),length(node));
UG = tril(G_rand + G_rand');
[shortest_dist,shortest_path,shortest_pred] = graphshortestpath(UG, OD_table);
shortest_path_transpose = shortest_path';

count_row = length(algo_path_point1);
algo_path_point1((count_row+1):(count_row+length(shortest_path_transpose)),:
algo_path_point1((count_row+1):(count_row+length(shortest_path_transpose)),3) = algo_path_point1(count_row+1,3);
algo_path_point1((count_row+1):(count_row+length(shortest_path_transpose)),4) = algo_path_point1(count_row+1,4);
for k = 1:length(shortest_path_transpose)
    algo_path_point1(count_row+k,3) = node(algo_path_point1(count_row+k,5), algo_path_point1(count_row+k,6));
end

for i = 1:length(algo_path_point1)
    temp_index_point = find(graph(:,3) == algo_path_point1(i,5) | graph(:,4) == algo_path_point1(i,5));
    if length(temp_index_point) > 0
        algo_path_point1(i,6) = graph(temp_index_point(1),2);
    else
        algo_path_point1(i,6) = 999;
    end
end

for i = 1:length(algo_path_point1)
    temp_index_point = find(graph(:,3) == algo_path_point1(i,5) | graph(:,4) == algo_path_point1(i,5));
    if length(temp_index_point) > 0
        algo_path_point1(i,6) = graph(temp_index_point(1),2);
    else
        algo_path_point1(i,6) = 999;
    end
end

temp_route = unique(algo_path_point1(:,1));
for i = 1:length(temp_route)
    for j = 1:100
        temp_point = []; 
        temp_point = find(algo_path_point1(:,1) == i & algo_path_point1(:,8) == j);
        for k = 1:length(temp_point)
            if k == length(temp_point)%%the last point for the path,mark as 99999
                algo_path_point1(temp_point(k),7) = 99999;
            else
                if length(find((algo_path_point1(temp_point(k),5) == graph(:,3) & a
                algo_path_point1(temp_point(k),7) = 0;
            else
                temp_link = find((algo_path_point1(temp_point(k),5) == graph(:,3)
                algo_path_point1(temp_point(k),7) = temp_link(1);
            end
        end
    end
end

* GIS Data Processing

%%find identical points according to their geographical location. Keep one, delete o
%%1:keep. 0:delete.
for i = 1:length(initial)
    if i == 1
        initial(i,5) = 1;
    else
        for j = 1:i-1
            if initial(i,3) == initial(j,3) & initial(i,4) == initial(j,4)
initial(i,5) = 0;
break;
else
    initial(i,5) = 1;
end
end
end

%%%find the useful data
final_row = find(initial(:,5)==1)

%%%duplicate the final data to the node table
for i = 1:length(final_row)
    node(i,1) = initial(final_row(i),1);
    node(i,2) = initial(final_row(i),2);
    node(i,3) = initial(final_row(i),3);
    node(i,4) = initial(final_row(i),4);
end

%%%rename the node ID using natural numbers
for i = 1:length(node)
    node(i,5) = i;
end

%%%node table: origin node ID, origin link, long, lat, new node ID
%%%link table: origin link ID, node 1, node 2, new link ID, road type
link(:,1) = unique(initial(:,2));
for i = 1:length(link)
    link(i,2) = find(initial(2*i-1,3)==node(:,3) & initial(2*i-1,4)==node(:,4));
    link(i,3) = find(initial(2*i,3)==node(:,3) & initial(2*i,4)==node(:,4));
end
rename the link ID using natural numbers
for i = 1:length(link)
    link(i,4) = i;
end

for road type: cycleway 1; footway 2; motorway 3; motorway_link 4; path
5; pedestrian 6; primary 7; primary_link 8; residential 9; secondary
10; secondary_link 11; service 12; steps 13; tertiary 14; trunk 15;
trunk_link 16; unclassified 17.

build graph

graph table: link ID, road type, node 1, node 2,
long(node1), lat(node1), long(node1), lat(node1)
dist, az, distance_meter
for i = 1:length(link)
    graph(i,1) = link(i,4);  
    graph(i,2) = link(i,5);  
    graph(i,3) = link(i,2);  
    graph(i,4) = link(i,3);  
    graph(i,5) = node(graph(i,3),3);  
    graph(i,6) = node(graph(i,3),4);  
    graph(i,7) = node(graph(i,4),3);  
    graph(i,8) = node(graph(i,4),4);
end

calculate the distance between two end points of a link
for j = 1:length(graph)
    [graph(j,9),graph(j,10)] = distance(graph(j,6),graph(j,5),graph(j,8),graph(j,7))
end

%%transfer the distance to the meter unit.
graph(:,11) = graph(:,9) * 1.1078e+005;

%%generate random distance using formula exp(random*number)*distance
rand_distance(1:length(graph),1) = exp(randn(length(graph),1)) .* graph(1:length(gr

%%build the random weighted network.
G_rand = sparse(graph(:,3),graph(:,4),rand_distance(:,1),length(node),length(node))

%%build the undirected network
UG = tril(G_rand + G_rand');
%%find the shortest path
[shortest_dist,shortest_path,shortest_pred] = graphshortestpath(UG,3079,4337,'direc

shortest_path_second = shortest_path';

for i = 1:length(shortest_path_second)
    shortest_path_second(i,2) = node(shortest_path_second(i,1),4);
    shortest_path_second(i,3) = node(shortest_path_second(i,1),3);
end

• Building OD Table

%%taxiID,revealed path, lat1,long1,lat2,long2,link number,distance,
%%primary, motorway,trunk,secondary,Node1,Node2,pathID

%%taxi1
no_of_path = unique(compact_path_point1(:,1));%%find the number of path
for i = 1:length(no_of_path)
index_of_temp_path = find(compact_path_point1(:,1)==i);
OD_table_taxi1(i,1) = 1; 1 for taxi1, 2 for taxi2, etc
OD_table_taxi1(i,2) = 1; 1 for revealed path, 0 for alternative path

OD_table_taxi1(i,3) = compact_path_point1(index_of_temp_path(1),3);
OD_table_taxi1(i,4) = compact_path_point1(index_of_temp_path(1),4); the origin

OD_table_taxi1(i,5) = compact_path_point1(index_of_temp_path(end),3);
OD_table_taxi1(i,6) = compact_path_point1(index_of_temp_path(end),4); the dest

OD_table_taxi1(i,7) = length(index_of_temp_path)-1; the link number of the path

temp_path_distance = 0; initialize
for j = 1:length(index_of_temp_path)-1
    [temp_dist,temp_az] = distance(compact_path_point1(index_of_temp_path(j,1),
    temp_dist_meter = temp_dist *1.1078e+005;
    temp_path_distance = temp_path_distance+temp_dist_meter;
end

OD_table_taxi1(i,8) = temp_path_distance; the distance of the path
end

for i = 1:length(OD_table_taxi1(:,1))
    OD_table_taxi1(i,9) = length(find(compact_path_point1(:,1)==i & (compact_path_p
    OD_table_taxi1(i,10) = length(find(compact_path_point1(:,1)==i & (compact_path_  
    OD_table_taxi1(i,11) = length(find(compact_path_point1(:,1)==i & (compact_path_  
    OD_table_taxi1(i,12) = length(find(compact_path_point1(:,1)==i & (compact_path_  
end

% build the final OD table
temp_path = unique(OD_table_algo_taxi1(:,13));
temp_row = 0;
for i = 1:length(temp_path)
    temp_path_index = find(OD_table_taxi1(:,15)==temp_path(i));
    temp_row = temp_row + 1;
    OD_table_final_taxi1(temp_row,:) = OD_table_taxi1(temp_path_index,:);
end

%%construct the final path and point table for revealed path

%%taxi1

temp_path = unique(OD_table_final_taxi1(:,15));
final_compact_path_point1 = [];
temp_row = 0;
for i = 1:length(temp_path)
    temp_path_point = find(compact_path_point1(:,1)==temp_path(i));
    temp_row = length(final_compact_path_point1);
    final_compact_path_point1(temp_row+1:temp_row+length(temp_path_point),:) = comp
end

%%taxiID, stated path, lat1, long1, lat2, long2, link number, distance,
%%primary, motorway, trunk, secondary, Node1, Node2,
%%routeID, pathID (here have 100 paths for each revealed path)

row_in_OD_algo = 0;%%initialize; the row index for each alternative path
no_of_route = unique(algo_path_point1(:,1));
for i = 1:length(no_of_route)%% loop for route
    i = no_of_route(i);
    for j = 1:100 %100 paths for each route
row_in_OD_algo = row_in_OD_algo + 1;

index_of_temp_path = find(algo_path_point1(:,1) == i & algo_path_point1(:,8) == OD_table_algo_taxi1(row_in_OD_algo,1)) == 1; 1 for taxi 1, 2 for taxi 2, etc
OD_table_algo_taxi1(row_in_OD_algo,2) = 0; 0 for revealed path, 0 for alt
OD_table_algo_taxi1(row_in_OD_algo,3) = algo_path_point1(index_of_temp_path)
OD_table_algo_taxi1(row_in_OD_algo,4) = algo_path_point1(index_of_temp_path)
OD_table_algo_taxi1(row_in_OD_algo,5) = algo_path_point1(index_of_temp_path)
OD_table_algo_taxi1(row_in_OD_algo,6) = algo_path_point1(index_of_temp_path)
OD_table_algo_taxi1(row_in_OD_algo,7) = length(index_of_temp_path)-1; 1; link

temp_path_distance = 0; distance initialize
for k = 1:length(index_of_temp_path)-1
    [temp_dist,temp_az] = distance(algo_path_point1(index_of_temp_path(k,1)
    temp_dist_meter = temp_dist *1.1078e+005;
    temp_path_distance = temp_path_distance+temp_dist_meter;
end
OD_table_algo_taxi1(row_in_OD_algo,8) = temp_path_distance;

OD_table_algo_taxi1(row_in_OD_algo,9) = length(find(algo_path_point1(:,1) ==
OD_table_algo_taxi1(row_in_OD_algo,10) = length(find(algo_path_point1(:,1) ==
OD_table_algo_taxi1(row_in_OD_algo,11) = length(find(algo_path_point1(:,1) ==
OD_table_algo_taxi1(row_in_OD_algo,12) = length(find(algo_path_point1(:,1) ==

OD_table_algo_taxi1(row_in_OD_algo,13) = i;
OD_table_algo_taxi1(row_in_OD_algo,14) = j;
%%keep the identical path and delete the same path

%%taxi1

temp_route = unique(OD_table_algo_taxi1(:,13));
OD_final_algo_taxi1 = [];
temp_row_in_final_OD_table = 0;

for i = 1:length(temp_route)
    temp_index_path = find(OD_table_algo_taxi1(:,13)==temp_route(i));
    for j = 1:length(temp_index_path)

        if j == 1%%the first line
            temp_row_in_final_OD_table = temp_row_in_final_OD_table + 1;
            OD_final_algo_taxi1(temp_row_in_final_OD_table,:) = OD_table_algo_taxi1
        else
            temp_index_path_norepeat = find(OD_final_algo_taxi1(:,13)== temp_route(temp_count_difference = 0;
            for k = 1:length(temp_index_path_norepeat)
                if OD_table_algo_taxi1(temp_index_path(j),7) == OD_final_algo_taxi1
                    temp_row_in_final_OD_table = temp_row_in_final_OD_table + 0;
                else
                    temp_count_difference = temp_count_difference + 1;
                end
            end
        end

    end

    if temp_count_difference == length(temp_index_path_norepeat) %% mea
        temp_row_in_final_OD_table = temp_row_in_final_OD_table + 1;
        OD_final_algo_taxi1(temp_row_in_final_OD_table,:) = OD_table_algo_taxi1
    end
end
%%%construct the final table for alternative paths
%%%taxi1
algo_final_path_point1 = [];
temp_route = unique(OD_final_algo_taxi1(:,13));
temp_row_in_algo_final = 0;
for i = 1:length(temp_route)
    temp_path = find(OD_final_algo_taxi1(:,13)==temp_route(i));
temp_path_index = OD_final_algo_taxi1(temp_path,14);
    for j = 1:length(temp_path_index)
        temp_point = [];
        temp_point = find(algo_path_point1(:,1)== temp_route(i) & algo_path_point1(temp_path_index(j),1))
        temp_row_in_algo_final = length(algo_final_path_point1);
        algo_final_path_point1((temp_row_in_algo_final+1):(temp_row_in_algo_final+1))
    end
end

%%%calculate the distance of the link
%%%taxi1
temp_route = unique(algo_final_path_point1(:,1));
for i = 1:length(temp_route)
    temp_path = find(algo_final_path_point1(:,1)== temp_route(i));
temp_path_index = unique(algo_final_path_point1(temp_path,8));

    for j = 1:length(temp_path_index)
        temp_path_point = find(algo_final_path_point1(:,1) == temp_route(i) & algo_
for \( k = 1: \text{length}(\text{temp}\_\text{path}\_\text{point}) \)

    if \( k == \text{length}(\text{temp}\_\text{path}\_\text{point}) \)
        \( \text{algo}\_\text{final}\_\text{path}\_\text{point1}(\text{temp}\_\text{path}\_\text{point}(k),9) = 0; \)
    else
        \([\text{temp}\_\text{dist}, \text{temp}\_\text{az}] = \text{distance}(\text{algo}\_\text{final}\_\text{path}\_\text{point1}(\text{temp}\_\text{path}\_\text{point}\text{temp}\_\text{distance} = \text{temp}\_\text{dist} \times 1.1078e+005;
        \text{algo}\_\text{final}\_\text{path}\_\text{point1}(\text{temp}\_\text{path}\_\text{point}(k),9) = \text{temp}\_\text{distance}; \)
    end
end
end
Bibliography


